

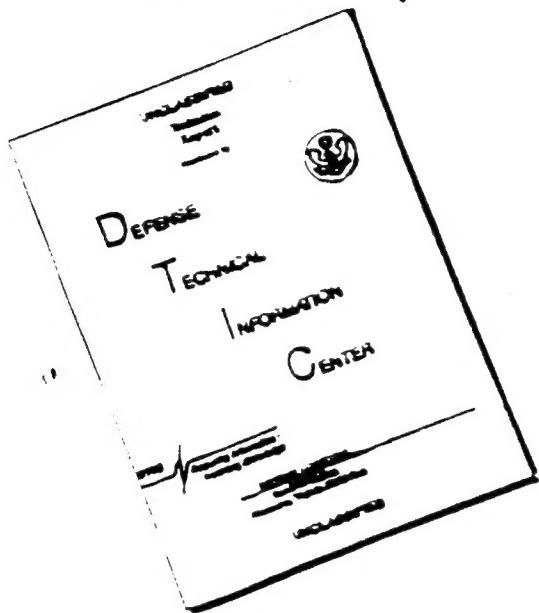
REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT	
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE		Approved for public release distribution unlimited	
4. PERFORMING ORGANIZATION REPORT NUMBER		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION <u>Rutgers, The State University of New Jersey</u>		6b. OFFICE SYMBOL (if applicable) <u>N/A</u>	7a. NAME OF MONITORING ORGANIZATION <u>Air Force Office of Scientific Research</u>
6c. ADDRESS (City, State, and ZIP Code) <u>New Brunswick, NJ 08903</u>		7b. ADDRESS (City, State, and ZIP Code) <u>AFOSR/NM, Building 410 Bolling AFB, DC 20332-6448</u>	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION <u>AFOSR</u>		8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER <u>AFOSR-91-0343</u>
8c. ADDRESS (City, State, and ZIP Code) <u>AFOSR/NM, Building 410 Bolling AFB, DC 20332-6448</u>		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO. <u>61102F</u>	PROJECT NO. <u>2304</u>
11. TITLE (Include Security Classification) <u>Mathematical Theory of Neural Networks -- Final Report (unclassified)</u>		TASK NO.	
12. PERSONAL AUTHOR(S) <u>Sontag, Eduardo D. and Sussmann, Hector J.</u>		WORK UNIT ACCESSION NO.	
13a. TYPE OF REPORT <u>Final Report</u>	13b. TIME COVERED <u>FROM 8/1/91 TO 7/31/94</u>	14. DATE OF REPORT (Year, Month, Day) <u>1994 8 31</u>	15. PAGE COUNT <u>11</u>
16. SUPPLEMENTARY NOTATION			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)			
<p>This report provides a summary of the grant work by the principal investigators in the area of neural networks. The topics covered deal with: (1) analysis of networks from the viewpoint of analog computational devices, exploring limitations imposed by resource constraints; (2) questions of parameter identification of recurrent nets; (3) use of feedforward nets for function approximation and interpolation problems; (4) systems theory (observability and other properties) for nets; and (5) the use of neural networks for the control of nonlinear systems.</p>			

19950403 054

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	<input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION <u>UNCLASSIFIED</u>
22a. NAME OF RESPONSIBLE INDIVIDUAL <u>Dr. Marc Jacobs</u>	22b. TELEPHONE (Include Area Code) <u>(202) 767-5027</u>	22c. OFFICE SYMBOL <u>NM</u>

DISCLAIMER NOTICE



**THIS DOCUMENT IS BEST
QUALITY AVAILABLE. THE COPY
FURNISHED TO DTIC CONTAINED
A SIGNIFICANT NUMBER OF
PAGES WHICH DO NOT
REPRODUCE LEGIBLY.**

MATHEMATICAL THEORY OF NEURAL NETWORKS

Final Technical Report, 1 August 1991 to 31 July 1994

Eduardo D. Sontag and Héctor J. Sussmann
SYCON – Rutgers Center for Systems and Control
Department of Mathematics, Rutgers University
New Brunswick, NJ 08903

Accession For	
NTIS	CRA&I <input checked="" type="checkbox"/>
DTIC	TAB <input type="checkbox"/>
Unannounced <input type="checkbox"/>	
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

ABSTRACT

This report provides a comprehensive, cumulative, and substantive summary of the progress and significant accomplishments achieved during the total period of the research effort covered by this contract. It deals with several topics related to the mathematical properties of feedforward as well as feedback nets, touching upon the following subareas: For feedforward nets: local minima, uniqueness of weights, architectures, classification capacity, uses for implementation of state-feedback control – especially for the control of linear systems with saturation, – approximation rates, and (PAC) learning issues. For recurrent nets: recurrent perceptrons, parameter identification and learning algorithms, systems-theoretic properties (observability, controllability), and theoretical computational capabilities.

1 Introduction

A huge amount of activity has taken place during the last few years in the area encompassed by the term "artificial neural networks," as evidenced by the proliferation of conferences, journals, electronic discussion groups, and patent applications. This area had seen many periods of hype (and justified reaction to that hype) since the late 1940s. While there was continuing activity by major researchers such as Kohonen and Grossberg all along, the latest resurgence in interest was due in large part to the appearance of two separate lines of work: (1) feedforward nets or "multilayer perceptrons" with sigmoidal activations* to be fit to data through nonlinear optimization, and (2) feedback nets, with a technique for associative memory storage and retrieval due to Hopfield. The impact of this work was amplified by the coincidental sudden easy availability of huge raw computational power, in amounts which were unimaginable when similar ideas had been considered in the past.

Main Motivation for This Work

Many practical successes of the associated technologies have been claimed in both the engineering and popular press. In the context of the AFOSR mission, one may mention a 1990 workshop centered on aerospace applications of neural nets, which took place at McDonnell Douglas Corp. and was attended by numerous AFOSR contractors (including the first PI) and researchers from several AF development labs. Presentations focused on the use of network techniques in flight systems design, such as fault detection and classification (and associated problems of reconfigurable aircraft control), development of controls valid over large flight envelopes, and even precision laying of composites on aircraft structures. A recurrent concern raised throughout the discussions during the workshop was the lack of theoretical foundations for the analysis and comparison of different network tools. The main goal of the work reported here is the research into fundamental theoretical issues which arise in the study of the capabilities and performance of neural networks. The PI's are mathematicians who are currently engaged in a program of research whose main purpose is to carry out a rigorous mathematical analysis for a number of problems in neural nets for which, so far, only heuristic methods have been developed. They believe that such a development, besides being intrinsically of interest, leads to a better understanding of the issues involved in the design of efficient algorithms as well as in understanding the possibilities and limitations of these models.

Another Motivation

There is another motivation as well that underlies the work described here, and is more basic than the understanding of neural net models in themselves. This arises from an issue that has come up in many domains—including computer, communications, and control engineering—concerning the interface between on the one hand the continuous, physical, world and on the other hand discrete devices such as digital computers, capable of symbolic processing. Lately the term "hybrid" systems has become popular in this context.

For instance, classical control techniques especially for linear systems have proved spectacularly successful in automatically regulating relatively simple systems; however, for large-scale problems, controllers resulting from the application of the well-developed theory are used as building blocks of more complex systems. The integration of these systems is often accomplished by means of ad-hoc techniques that combine pattern recognition devices, various types of switching controllers, and humans

*Contrary to popular misconceptions, multilayer nets—but with discontinuous activations—had already been much studied in the widely unread but widely cited early 1960s book by Rosenblatt, and feedback nets were studied by him as well.

—or, more recently, expert systems.— in supervisory capabilities. This has caused renewed interest in the formulation of mathematical models in which the interface between the continuous and the symbolic is naturally accomplished and system-theoretic questions can be formulated and resolved for the resulting models. Successful approaches will eventually allow the interplay of modern control theory with automata theory and other techniques from computer science. This interest has motivated much research into areas such as discrete-event systems, supervisory control, and more generally “intelligent control systems”. The present work has as one of its underlying objectives the analysis of neural networks as a paradigm in which to understand many of these issues. Other paradigms could be used as well; it so turns out that neural nets are a particularly appealing and extremely natural class of nonlinear systems.

Similarly, in numerical analysis and optimization, much activity has taken place in the realm of continuous algorithms. Included here are such areas as differential-equation implementations of “interior methods” for linear and nonlinear programming, the use of flows on manifolds to solve eigenvalue and optimization problems (in particular the work of Brockett and his school) and the Blum-Shub-Smale approach to “real valued” algorithms. All these deal in one way or another with the power of “analog computing” to solve problems that can also be attacked with discrete/symbolic techniques. Again, we view the neural net paradigm as one in which to explore the interface between “digital” and “analog” modes of computation. In fact, recent work by one of the PIs and his students has succeeded in formulating a new computer-science approach to these issues, as will be described in the report.

Why Neural Nets?

Motivating the use of nets is the belief that in some sense they are an especially appropriate family of parameterized models, as opposed to, say, finite Fourier series or splines. Typical engineering justifications range from parallelism and fault tolerance to the possibility of using hardware implementation. Numerical and statistical advantages are said to include excellent capabilities for “learning,” “adaptation,” and “generalization.” We do not argue the case for or against this belief here. Notwithstanding some popular claims to the contrary (for instance, supported by recent rate of approximation results, a topic which is reviewed in the report) we feel that the situation is still very unclear regarding the relative merits of using neural nets. In any case, it is quite likely that techniques based on neural networks will play some role as part of the general set of tools in estimation, learning, and control. As mathematicians, we are interested at this stage in obtaining a deeper understanding of the topic, as a prerequisite to theoretical comparisons.

Scope and Organization of the Report

The term “neural network” tends to be applied loosely, making the area too broad and ill-defined. For purposes of this report, however, we adopt the most popular paradigm: (artificial) neural nets are systems composed of saturation-type nonlinearities, linear elements, and optionally dynamic components (integrators in continuous time, delays in discrete time). While still a huge subject, this definition leaves out a variety of topics sometimes included, such as “radial basis” networks or multiplicative effects. (By narrowing-down the area we can get stronger general mathematical results, but no value judgments are implied regarding the interest of such alternative structures.) It is possible to organize the topic, once it is so defined, by classifying nets into broad categories, and studying separately different mathematical questions that are natural for each subclass.

The topics to be discussed will be organized roughly into these broad categories:

- No Dynamics (“Feedforward Nets”)

- Local Minima and Uniqueness of Weights
 - Architectures and Classification Capacity
 - Uses for Implementation of State-Feedback Control
 - Single vs Multiple Layers
 - Control of Linear Systems with Saturation
 - Approximation Rates
 - Learning
- Linear Dynamics (“Recurrent Perceptrons”)
 - Nonlinear Dynamics (“Fully Recurrent Nets”)
 - Parameter Identification, Observability, Controllability.
 - Computational Capabilities

The next Part provides a brief outline of the above areas, with as few technical details as sufficient to convey the main issues and results. The last Part provides some more details on selected subtopics, but overall most of the discussion in this report is informal, with references to the literature for precise technical points.

2 Overview

As discussed above, by an artificial neural net we mean here a system which is as an interconnection of basic processors, each of which takes as its inputs the outputs from other processors as well as from outside the system, and performs a certain nonlinear transformation $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ (the “activation function”) on an affine combination of these inputs. For simplicity, we will assume that each processor uses the same transformation σ . The coefficients of the affine combination, or “weights,” of course vary from processor to processor. The output signal produced by each such basic processing element is broadcast to the rest of the system; some of the signals are combined into the output of the whole system. Thus the basic unit for interconnections is as in Figure 1.

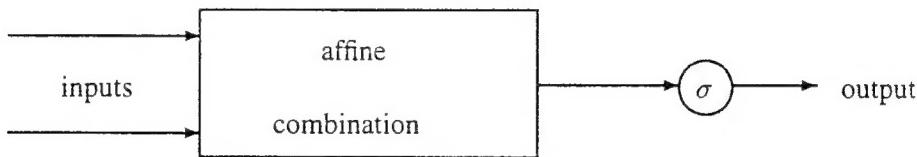


Figure 1: *Basic Processing Element*

The interconnection structure is often called the *architecture* of the net; together with the choice of σ and the values for weights—which are typically thought of as “programmable” parameters and are the subject of numerical optimization—it determines completely the behavior of the network; later we of course give more precise definitions.

One of the main functions σ typically used is $\text{sign}(x) = x/|x|$ (zero for $x=0$), or its relative, the hardlimiter, threshold, or *Heaviside* function $H(x)$, which equals 1 if $x > 0$ and 0 for $x \leq 0$ (in either case, one could define the value at zero differently, results do not change much). In order to apply various numerical techniques, one often needs a differentiable σ that somehow approximates $\text{sign}(x)$ or $H(x)$. For this, it is customary to consider the hyperbolic tangent $\tanh(x)$, which is close to the sign function when the “gain” γ is large in $\tanh(\gamma x)$. Equivalently, up to translations and change of coordinates, one may use the *standard sigmoid*

$$\sigma_s(x) = \frac{1}{1 + e^{-x}}.$$

Also common in practice is a piecewise linear function,

$$\pi(x) = \begin{cases} -1 & \text{if } x \leq -1 \\ 1 & \text{if } x \geq 1 \\ x & \text{otherwise.} \end{cases}$$

This is sometimes called a “semilinear” or “saturated linearity” function. See Figure 2.

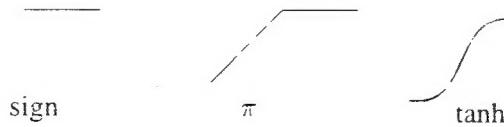


Figure 2: *Different Functions σ*

Whenever time behavior is of interest, one also includes dynamic elements, namely delay lines if dealing with discrete-time systems, or integrators in the continuous-time case, so that a well-defined dynamical system results.

Static nets are those formed by interconnections without loops –otherwise the behavior may not be well-defined– and are also called *feedforward* nets, in contrast to the terms *feedback*, *recurrent*, or *dynamic* nets used in the general case. Figure 3 provides “block diagrams” for the examples of a static net computing the function

$$y = 2\sigma[3\sigma(5u_1 - u_2) + 2\sigma(u_1 + 2u_2 + 1) + 1] + 5\sigma[-3\sigma(u_1 + 2u_2 + 1) - 1]$$

and a continuous-time dynamic net representing the system

$$\dot{x}_1 = \sigma(2x_1 + x_2 - u_1 + u_2), \quad \dot{x}_2 = \sigma(-x_2 + 3u_2), \quad y = x_1.$$

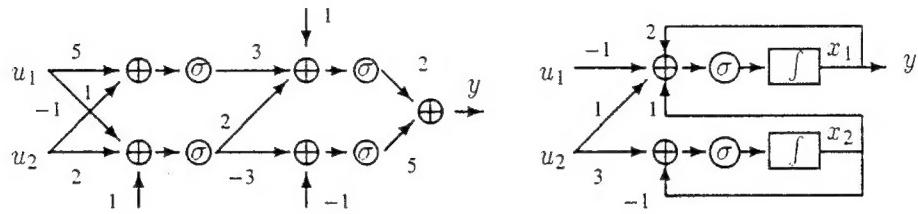


Figure 3: Examples of one Feedforward and one Recurrent Net

Mathematically we think of feedforward nets as computing functions between finite-dimensional input and output spaces; for instance, in the first example in Figure 3 the net induces a mapping $\mathbb{R}^2 \rightarrow \mathbb{R}$. Dynamic nets, in contrast, are a subclass of systems in the sense of control theory, and must be defined in terms of differential or difference equations with inputs and outputs. They give rise to mappings between function spaces. For instance, the second example in Figure 3 could be seen as inducing a mapping between pairs of measurable functions $(u_1(t), u_2(t))$ and continuous functions $y(t)$ defined for $t \geq 0$ (assuming a fixed initial state such as $x_1(0)=x_2(0)=0$ and assuming sufficient regularity, for instance that σ is a globally Lipschitz function). Feedforward nets are used for pattern recognition and classification, and as approximators of functions, and appear for instance when implementing static feedback laws for certain controlled systems. Dynamic nets provide an appealing class of nonlinear dynamical systems, and have been used in implementations of classifiers for problems involving correlated temporal data, as well as universal models for adaptive control. Some of these roles will be discussed more later.

An intermediate case of interest between feedforward nets and the full feedback structure is given by what the PIs call *recurrent perceptrons* by analogy with the perceptrons that have classically been studied in the area. In this case, the dynamical behavior is linear but there is a nonlinearity at the output. This special case will be discussed separately.

Local Minima and Weight Uniqueness for Feedforward Nets

Typical applications of neural nets are to binary classification, interpolation, and function approximation problems. For any given fixed architecture (and choice of activation function), the free parameters (weights) are adjusted in order to better approximate a function or attain a classification objective. During a training or supervised learning stage, labeled examples are presented, and parameters are adjusted so as to make the network’s numerical output close to the desired values. A steepest-descent (or elaborations thereof) minimization of an error criterion is often used at this point. Later, during actual operation, the output given by the net when a new input is presented will be taken as the network’s guess at the “right” value. Presumably, one advantage of using nets in this mode is that the parallel processing structure of nets means that this guess could be computed fairly fast in special-purpose hardware.

One may attempt to justify the use of such models for prediction applications on the basis of computational learning theory –see below– but most empirical work uses statistical cross-validation techniques for this purpose. Whatever the justification, a major set of issues to be addressed centers on the fact that the procedure involves a hard nonlinear minimization problem. Thus, understanding the structure of the set of local minima of the error function is a must. In early work in this area by the PIs and others, the existence of nonglobal local minima was rigorously shown to happen in even the simplest possible architectures. For a special case, a recasting in terms of “nonstrict” error functions results in a problem which does not suffer from spurious local minima, and for which gradient descent can be shown to converge globally (see [1]), but in the general case the existence of such obstructions is unavoidable.

Moreover, and related to this, many weights could potentially give rise to the same input/output behavior, and this multiplicity will obviously affect the optimization algorithm, since then different weights give the same cost. Hence, one must ask what the group of behavior-preserving weight transformations looks like. This question was posed in the late 1980s by Hecht-Nielsen. An answer was provided with a general result characterizing the possible symmetries for the general case of *one hidden layer (1HL)* nets –the most widespread architecture– by one of the PIs, in [4]. Such nets are those which diagrammatically can be represented as:

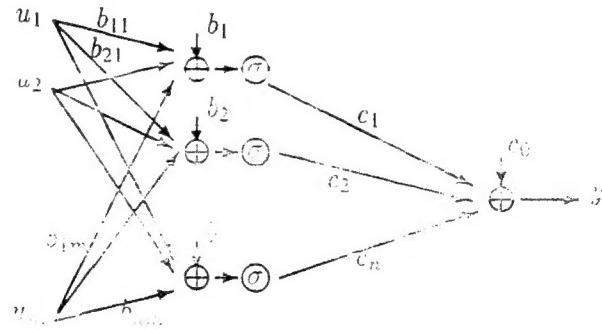


Figure 4: *One Hidden Layer (1HL) Net*

Here one deals with functions that can be expressed in the form of an affine combination of terms each of which consists of the activation σ applied to an affine combination of inputs:

$$c_0 + \sum_{i=1}^n c_i \sigma(B_i u + b_i) \quad (1)$$

where B_i is the i th row of the matrix B of interconnection coefficients from inputs u_i to the “hidden layer.” In more compact notation, the function represented is $f(u) = C\tilde{\sigma}_n(Bu + b) + c_0$, where $\tilde{\sigma}_n : \mathbb{R}^n \rightarrow \mathbb{R}^n$ indicates the application of σ to each coordinate of an n -vector:

$$\tilde{\sigma}_n(x_1, \dots, x_n) = (\sigma(x_1), \dots, \sigma(x_n)). \quad (2)$$

(The subscript n will be omitted as long as its value is clear from the context. Also, these notations are used if there are vector instead of scalar outputs y , in which case C is a matrix and c_0 a vector.) In other words, the behavior of a 1HL network is a composition of the type $f \circ \tilde{\sigma} \circ g$, where f and g are affine maps. In the case when u is a scalar ($m=1$), one is studying the span of dilates and translates of σ , somewhat in the spirit of wavelet theory. The interest is in understanding how many possible such representations there may exist for a given function.

The result in [4] established the finiteness of the group of symmetries under obvious generic assumptions –for instance, if some c_i vanishes, the corresponding B_i cannot be unique– for the special case of the activation tanh. The result is that invariance only holds under the obvious transformations: permutations and sign reversals in each term. The proof is based on asymptotic analysis techniques. In work of the other PI and coworkers ([17]) the result was extended to a wide class of real-analytic functions, using residue arguments. (The issue of weight symmetries has also been studied for recurrent nets, by the PIs and students, as mentioned later.) Together with results guaranteeing that a small number of samples is enough in order to determine the external behavior, as shown in the recent paper [33] as a consequence of results on stratification of subanalytic sets, this means that the map that evaluates the function at sufficiently many samples has generically discrete fibers. Together with recent results in logic that apply to tanh (again see [33]), this provides results on generic structure of local minima sets for the error function.

Architectures and Classification Capacity

It is essential to understand how many units are needed in order to solve a given classification or interpolation task. One of the PIs has worked substantially on such issues (see e.g. [3]), and this will be described in the report.

A closely connected question concerns the study of the relative advantages of different architectures. In particular, it is by now well-known, thanks to the pioneering work of Cybenko, Hornik, White, Lesnno, and others, that 1HL networks have universal approximation properties, assuming very little on the activation σ besides it not being polynomial (for simple proofs, which apply for restricted but yet quite general activations, see [18]). These results are stated in terms of density properties in spaces of continuous functions or in spaces of integrable functions with L^p norms, $p < \infty$. Moreover, there are encouraging results on approximated rates, as discussed later.

However, the quality of approximation with such 1HL architectures may not be as good, for fixed resources (e.g., number of processors), as those achievable with more complex architectures. For instance, the characteristic function of a square in the plane can be easily approximated by a *two hidden layer* (2HL) net, but good approximations using 1HL nets require many terms. More generally, one can make statements for classes well-approximated by certain multivariate splines. A very related disadvantage of 1HL vis a vis 2HL nets arises from the topologies in which the 1HL approximation theorems hold. One of the PIs, in [2], gave a general theorem that deals with this issue. A basic problem is that uniform approximation of discontinuous functions is in general impossible using 1HL nets: technically, there is no density on L^∞ , even on compact subsets, and an even stronger negative result holds, regarding the impossibility of constructing sections of certain coverings. This has serious implications in solving inverse problems, such as inverse kinematic computations in robotics, and indeed has been noted experimentally by many authors. The same problem appears in control applications, as explained later.

Implementations of State-Feedback Control; Saturated Actuators

Among the most popular areas of application for neural networks is that of control (see for instance the survey [18]). This part of the report touches upon various aspects of that topic.

One line of recent –but already widely cited– work of the PIs deals with the use of networks for the control of linear systems subject to actuator saturation,

$$\dot{x} = Ax + B\sigma(u)$$

where A and B are as usual in linear control theory and σ is a saturation such as \tanh or π . It is often said that saturation is the most commonly encountered nonlinearity in control engineering, so the development of techniques for the control of such systems is obviously of great interest. Our work starts with the result by Fuller, around 1970, that it is in general impossible to globally stabilize the origin of such systems by means of linear feedback $u=Fx$ –for a more general result along those lines, see [23]— even if the system is open-loop globally controllable to the origin. This suggests the obvious question of searching for *nonlinear* feedback laws $u=k(x)$ that achieve such stabilization. The interest here is in nicely behaved and easily implementable controllers, in contrast to optimal control techniques, which result in highly irregular feedback.

In 1990 work the PIs proved that *smooth* stabilization is always possible. Motivated by our paper, Teel showed that single-input multiple integrators can be stabilized by feedbacks which are themselves compositions of linear functions and iterated saturations. We were then able to extend Teel's result to arbitrary linear systems as above; see [26]. This is very satisfying mathematically: the control of a linear system involving saturation is achieved by a feedback law built—in a fairly simple manner—out of similar components. Pursuing this research further, we have now arrived at the rather surprising conclusion that one may always use the simplest possible nonlinear architecture, namely 1HL nets. (See [39] for a summary; the full paper will appear in [11]). As an application, the paper [34] describes an explicit example of control design for an F-8 aircraft subject to elevator rate constraints. Much needs to be done in this area, as our results provide unacceptable performance, but the improvement with respect to linear feedback is remarkable.

Thus, 1HL nets are found to be useful in controlling certain nonlinear systems, which is consistent with the designs proposed in much of the neurocontrol literature. The availability of a simple structure with clearly identifiable and tunable parameters is extremely appealing from an adaptive control point of view. Moreover, the universality results for function approximation would seem to indicate that such architectures are always sufficient. However, this is very misleading. It turns out that 1HL nets are not “universal” enough for the implementation of controllers, in a precise sense reviewed later. This was first pointed out in the paper [2], which also gave a general theorem showing that two hidden layers (and discontinuous activations) *are* sufficient. (The contribution was recognized with an honorable mention for outstanding paper in the IEEE Transactions on Neural Networks for 1992.)

Intuitively, the reason for this apparent contradiction is that control problems are essentially inverse problems (one must solve for a trajectory satisfying certain boundary conditions). For such, as pointed out above, more general architectures are needed. To give an outline of how this obstruction can occur, consider the following situation. Suppose that the objective is to globally asymptotically stabilize a planar system

$$\dot{x} = f(x, u)$$

with respect to the origin using controllers $u=k(x)$ implementable by 1HL nets. It is easy to give examples for which there is no continuous feedback law that will achieve stabilization, even if the original system is very simple. So no 1HL net with continuous activations would be able to accomplish the stated objective. But what about allowing discontinuous σ such as Heaviside activations? (We ignore for now questions of possible nonexistence of solutions for ode's with discontinuous right-hand sides; the problem will be even more basic.) It turns out that even then it may be impossible to stabilize. Indeed, assume that we know some discontinuous feedback law $k_0(x)$ which stabilizes. It would appear that one can then obtain $k(x)$ simply by approximating k_0 . However, as we said in an earlier section, it is impossible in general to approximate a discontinuous k_0 uniformly by 1HL functions (this is an easy result pointed out in [2]). A weak type of approximation may not be enough for control purposes. For instance, it may be the case that for each approximant k_0 there is some smooth simple closed curve Γ encircling the origin where the approximation is bad and that this causes the vector field $f(x, k_0(x))$

to point transversally outward everywhere on Γ ; in that case trajectories cannot cross Γ . Thus a bad approximation in a very small set (even of measure zero) can introduce a “barrier” to global stabilization.

The paper [2], for discrete-time, constructs examples of systems which are otherwise stabilizable but such that every possible feedback implementable by a 1HL net (with basically any type of activation, continuous or not) must give rise to a nontrivial periodic orbit. On the other hand, it can be shown that every system that is stabilizable, by whatever k , can also be stabilized using 2HL nets with discontinuous activations (under mild technical conditions, and using sampled control). See [2] for details.

To summarize, if stabilization requires discontinuities in feedback laws, it may be the case that no possible 1HL net stabilizes. Thus the issue of stabilization by nets is closely related to the standard problem of continuous and smooth stabilization of nonlinear systems, one that has attracted much research attention in recent years. Roughly, there is a hierarchy of state-feedback stabilization problems: those that admit continuous solutions, those that don’t but can still be solved using 1HL nets with discontinuous activations, and more general ones (solvable with 2HL). It can be expected that an analogous situation will be true for other control problems. Perhaps the reason that most neurocontrol papers have reported success while using 1HL nets is that they almost always dealt with feedback linearizable systems, which form an extremely restricted class of systems that happen to admit continuous stabilizers.

Approximation Rates

Certain recent results due to Andrew Barron and Lee Jones have been used to support the claim that (1HL) neural network approximations may require less parameters than conventional techniques. What is meant by this is that approximations of functions in certain classes (defined typically in harmonic analysis terms or by bounds in suitable Sobolev norms) to within a desired error tolerance can be obtained using “small” networks. In contrast, the argument goes, using for instance orthogonal polynomials, splines, or Fourier series, would require an astronomical number of terms, especially for multivariate inputs. Unfortunately, this claim represents a misunderstanding of the very nice contributions of Barron and Jones. Their results apply in principle also to give efficient approximations with various types of classical basis functions, *as long as the basis elements can be chosen in a nonlinear fashion*, just as with neural networks. For instance, splines with varying (rather than fixed) nodes, or trigonometric series with adaptively selected frequencies, will have the same properties. What is important is the possibility of *selecting* terms adaptively, in contrast to the use of a large basis containing many terms and fitting these through the use of least squares. Thus, the important fact about the recent results is that they emphasize that *nonlinear* parameterizations may require less parameters than linear ones to achieve a guaranteed degree of approximation. (Abstractly, this is not so surprising: for an analogy, consider the fact that a one-parameter analytic curve, based on ergodic motions, can be used to approximate arbitrarily well every element in an Euclidean space \mathbb{R}^d , but no $(d - 1)$ -dimensional subspace can do so.) For an exposition and a precise theorem comparing rates of approximation by such neural net and nonlinear adaptation approaches vs. rates obtainable with classical approximation techniques see the paper [15].

The Barron-Jones results are mostly for least-squares approximation (though Barron did provide an L^∞ result as well). In robust estimation, the advantages of norms different from quadratic are well-known. This motivated the work [37], [14], which established good rates of approximation when measuring errors in several L^p norms as well as limitations of the “greedy” or “incremental” technique suggested by Jones in his study of neural nets and projection-pursuit estimation. The work in [37], [14] is based on ideas from the theory of stochastic processes on function spaces and techniques related to moduli of smoothness in Banach spaces.

Learning

The use of networks for pattern classification and related applications makes imperative the theoretical study of the question of learning. One of the main current approaches to defining and understanding the meaning of “learning” is based on the *probably approximately correct* (“PAC”) model proposed in computational learning theory by Valiant in the early 1980s. Very closely related ideas appeared in statistics even earlier, in the work of Vapnik and Chervonenkis and the interactions between statistics and computer science are the subject of much current research; for a quick survey of some basic results, see for instance [18].

In the PAC paradigm, a “learner” has access to data given by a labeled sample generated at random, with inputs independently and identically distributed according to some fixed but unknown probability measure. It is assumed that there is some fixed but unknown function generating the data, and this function belongs to some known class of functions which is used to characterize the assumptions (“bias”) being made about what is common among the observed input/output pairs. The learner knows the class but not the particular function generating the data. (For instance, in linear dynamical systems identification, the class could be that of all stable SISO systems of a certain McMillan degree.) The learner’s objective is to use the information gathered from the observed labeled samples in order to guess the correct function in this class, or, more precisely, to make a good guess as to the correct output for unseen inputs. (This can all be formulated elegantly in terms of L^1 norms. Also, note that many variations are possible, for instance allowing a “hypothesis” which is in a different class, allowing for “queries” by the learner, assuming more information on the probability distribution generating the samples, and so forth.) Learnability can then be defined in an *information-theoretic* sense, in terms of the requirement that a relatively small number of samples be generally sufficient to obtain a good approximation of the unknown function, or in a *complexity-theoretic* sense, requiring that in addition the amount of computation needed to actually make a prediction be relatively small.

For binary functions (that is, classification problems) and the information-theoretic question, the situation can be well-characterized in terms of the *Vapnik-Chervonenkis (VC) dimension* of the class of functions. Finiteness of this measure is necessary and sufficient for PAC learnability. For networks with *discontinuous*—more precisely Heaviside—activations, finiteness of VC dimension has been known for a long time, based on the work of Haussler and Baum. However, this did not apply to nets as implemented in practice, as differentiability is required for gradient descent algorithms. Recently, in the joint work [33], we solved the long-standing open question of showing finiteness of VC dimension, and hence learnability in the information-theoretic sense, for the more realistic case of sigmodal nets using activations \tanh and other standard activations. The results are based on very recent work in logic and show also the finiteness of VC dimension for multivariate sparse polynomials, which had been an open question in theoretical computer science.

Regarding complexity-theoretic aspects of learning, it was known since work of Blum and Rivest several years ago that with Heaviside activations the problem is not learnable (this amounts to showing NP-hardness of the “loading problem”) but for continuous activations, and in particular again for the activations used in practice such as \tanh , the question is still open. (There have been results announced in late 1993 applying to binary inputs, but for real-valued inputs the question is not settled.) Recent research (see [13]) has made partial progress towards that goal.

Linear Dynamics (“Recurrent Perceptrons”)

We now turn to recurrent nets, that is, the case in which the interconnection graph has loops and so a dynamic interpretation of behavior is necessary. A degenerate case is that in which no loop contains a

nonlinear element σ . Such systems are described by cascades of a linear system –in the standard sense in control theory– with a feedforward net. The simplest case is given by a difference or a differential equation

$$x(t+1) \text{ [or } \dot{x}(t)] = Ax(t) + Bu(t), \quad y(t) = \sigma_p(Cx),$$

in discrete- or continuous-time respectively (dot indicates time derivative), where A is an $n \times n$ matrix, B is $n \times m$, and C is $p \times n$. Here σ_p is a map which in each coordinate is a saturation-type nonlinearity, such as sign, π , or σ_s . We call such systems, for which the dynamics are linear but the outputs are subject to the limitations of measuring devices, *constrained-output (linear) systems*.

There are many reasons for studying such objects besides neural networks. As mentioned earlier, there is a generally recognized interest in understanding the continuous/discrete interface; one natural first step is the study of partial (discrete) measurements on the state of a continuous dynamical system, in the style of symbolic dynamics. One of the first questions that one may address concerns the nature of the information that can be deduced by a symbolic “supervisor” from data gathered from such a “lower level” continuous device, using appropriate controls in order to obtain more information about the system. Most sampling involves some form of quantization into a discrete range; a special case of this, 1-bit quantization, is that in which σ simply takes the sign of each coordinate, giving rise to *sign-linear (SL) systems*. These were the focus of the paper [6] by one of the PIs and a graduate student, which provided a complete necessary and sufficient characterization for the state-observability of SL systems, expressed in terms of algebraic rank conditions similar to the linear case. The results in that paper and related work by the same authors parallel those in the linear case, but with some, perhaps unexpected, differences. For instance the characterization is different in the continuous- and discrete-time cases, and controllability properties affect observability. In the more recent work [12], the same questions were asked for the case of *output-saturated* systems, those for which in each coordinate $\sigma = \pi$, that is, the measurement device is saturated for large values but is linear near zero. In this case, an elegant characterization is still not available for the case in which there are three or more output channels.

Sign-linear systems are also motivated by pattern recognition applications. Indeed, among the most popular techniques for that purpose are those based upon *perceptrons* or linear discriminants. Mathematically, these are simply functions of the type $\theta : \mathbb{R}^k \rightarrow \mathbb{R}^p$, $\theta(v) = \text{sign}(Cv)$ (sign is taken in each coordinate), typically with $k \gg p$. Perceptrons are used to classify input patterns $v = (v_1, \dots, v_k)$ into classes, and they form the basis of many statistical techniques. In many practical situations, arising in speech processing or learning finite automata and languages, the vector v really represents a finite window

$$u(t-1), \dots, u(t-s) \tag{3}$$

of a sequence of m -dimensional inputs $u(1), u(2), \dots$, where the components of (3) have been listed as v (and $sm = k$). In that case, the perceptron can be understood as a sign-linear system of dimension n , with a shift-register used to store the previous s inputs (3). Soon in our context, perceptrons are nothing more than the very special subclass of “finite impulse response” SL systems (all poles at zero). As such, they are not suited to modeling time dependencies and recurrences in the data. It is more reasonable to generalize to the case where a convolution by an arbitrary realizable kernel is taken before the sign, and this motivated the introduction of SL systems in the engineering and neural nets literature. (Models of this form also arise in many other areas as well. For instance, in signal processing, when modeling linear channels transmitting digital data from a quantized source, the channel equalization problem becomes one of systems inversion for systems that are essentially of this type, with quantizer σ .) We called the resulting input/output behaviors *sign-linear i/o maps*. For these, sign-linear systems constitute the most obvious class of state-space realizations. Thus one is interested in questions such as minimal dimension representations and the existence of Nerode-canonical (i.e., reachable and observable) realizations.

The paper [6] treated such issues, and in particular showed that canonical realizations admit a natural structure as cascades of linear systems and finite automata.

Recurrent Nets: Models

The general case of *recurrent* nets considered in this report consists of those systems evolving in \mathbf{R}^n according to equations of the type

$$x(t+1) \text{ [or } \dot{x}(t)] = \vec{\sigma}_n(Ax + Bu), \quad y = Cx. \quad (4)$$

Any such system is specified by providing a triple of matrices A, B, C and the activation function σ . We use notations consistent with linear systems theory (when σ is the identity), that is, A, B , and C are respectively real matrices of sizes $n \times n$, $n \times m$ and $p \times n$. See the block diagram in Figure 5, where $\Delta x = x^+$ or $= \dot{x}$ in discrete or continuous time respectively:

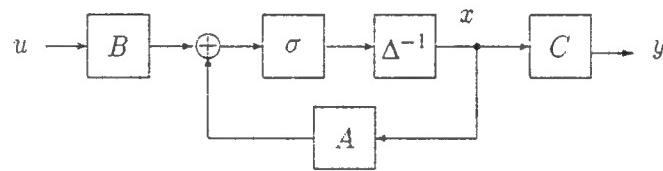


Figure 5: Recurrent Net

Some variations of the above model are also of interest. For instance one finds in the literature systems of the form

$$\dot{x} = -Dx + \vec{\sigma}(Ax + Bu)$$

with D a diagonal matrix (for “Hopfield nets” one picks in addition A symmetric). Also, the input term Bu may be outside the nonlinearity. For purposes of this report, we restrict attention to the form (4), but [7] shows how to transform among different models.

Recurrent nets have been studied for a long time, and appeared early on in the work of Hopfield as well as among the models considered by Grossberg and his school. They are sometimes interpreted as representing the evolution of ensembles of n “neurons,” where each coordinate x_i is a real-valued variable which represents the internal state of the i th neuron, and each $u_i, i = 1, \dots, m$, is an external input signal. The coordinates of $y(t)$ represent the output of p probes, or measurement devices, each of which averages the activation values of many neurons. (In much of the literature, C is just a projection on some coordinates, that is, the components of y are simply a subset of the components of x .)

With feedback, one may exploit context-sensitivity and memory, characteristics essential in the modeling and control of processes involving dynamical elements. Recurrent networks have been employed in the design of control laws for robotic manipulators (Jordan), as well as in speech recognition (Fallside, Kuhn), speaker identification (Anderson), formal language inference (Giles), and sequence extrapolation for time series prediction (Farmer). In signal processing and control, recurrent nets have been proposed as generic identification models or as prototype dynamic controllers. In addition, as discussed later, recent theoretical results about neural networks established their universality as models for systems approximation as well as analog computing devices.

Electrical circuit implementations of recurrent nets, employing resistively connected networks of n identical nonlinear amplifiers, with the resistor characteristics used to reflect the desired weights,

have been suggested as analog computers, in particular for solving constrained optimization problems and for implementing content-addressable memories. Special purpose analog chips are being built to implement recurrent nets directly in hardware.

However, and in spite of their attractive features, recurrent networks have not yet attained as much popularity as might have been expected, compared with the feedforward nets so ubiquitous in current applications. Perhaps the main reason for this is the fact that training (“learning”) algorithms for recurrent nets suffer from serious potential limitations. The learning problem is that of finding parameters that “fit” a general form to training (experimental) data, with the goal of obtaining a model which can subsequently be used for pattern recognition and classification, for implementation of controllers, or for extrapolation of numerical values. Various learning methodologies for recurrent networks have been proposed in the literature, and have been used in applications. All algorithms are based on an attempt to achieve the optimization of a penalty criterion by means of steepest descent, but due to memory and speed constraints, they usually only involve an estimate of the gradient. They differ on the approximation used, and thus on their memory requirements and convergence behavior. Among these are “recurrent backpropagation,” “backpropagation through time,” and the “real time recurrent learning” algorithm. Besides an imperfect approximation of the gradient, there is of course the issue of spurious local minima, as in the feedforward case. Part of this report deals with a new and totally different approach –based on recent theoretical results– for the identification of recurrent models. This brings us to the next point.

Parameter Identification, Observability, Controllability

As for feedforward nets, one can ask about weight uniqueness. In [7] (for continuous-time) and [35] (for discrete-time) one of the PIs and a graduate student studied to what extent does the function of the network –that is, the “black box” behavior of applying given input-output signals—determine the parameters, that is, the entries of the matrices A , B , and C . A more precise formulation is as follows. Assume that the network is started at the relaxed state $x(0)=0$ (nonequilibrium initial states are the subject of a paper in preparation) and an input signal $u(\cdot)$ is applied. Let $x(\cdot)$ be the solution of (4) –in continuous-time, assume that σ is globally Lipschitz so $x(t)$ is well-defined for all times– and let $y(t) = Cx(t)$ be the output signal thus generated. In this manner, for each triple (A, B, C) one defines an *input-output mapping* $\lambda_{(A,B,C)} : u(\cdot) \mapsto y(\cdot)$. The formal question is to what extent are the matrices A, B, C determined by the i/o mapping $\lambda_{(A,B,C)}$.

In the very special case when σ is the identity, classical linear realization theory implies that, generically, the triple (A, B, C) is determined only up to an invertible change of variables in the state space. That is, except for degenerate situations that arise due to parameter dependencies (non-controllability or non-observability), if two triples (A, B, C) and $(\hat{A}, \hat{B}, \hat{C})$ give rise to the same i/o behavior then there is an invertible matrix T such that $T^{-1}AT = \hat{A}$, $T^{-1}B = \hat{B}$, and $CT = \hat{C}$. This is the same as saying that the two systems are equivalent under a linear change of variables $x(t) = T\hat{x}(t)$. Conversely, still in the classical case $\sigma = \text{identity}$, any such T gives rise to another system with the same i/o behavior, when starting with any given triple (A, B, C) .

These classical facts essentially apply only when σ is linear. In [7] we showed that for nonlinear activations –under a mild nonlinearity axiom– the natural group of symmetries is far smaller than that of arbitrary nonsingular matrices. It is instead just a finite group, the “sign-permutation subgroup” given by the action of matrices T as above but having the special form of a permutation matrix composed with a diagonal matrix performing at most a sign reversal at each neuron. That is, *the input/output behavior uniquely determines all the weights, except for a reordering of the variables and, for odd activation functions, sign reversals of all incoming and outgoing weights at some units*. As for linear systems,

suitable genericity conditions, given by certain simple algebraic inequalities, are assumed. (The paper also shows that it is possible to determine σ itself from the i/o data as well.) This result has the surprising implication that a dimensionality reduction of the parameter space –as is the case with linear systems, where canonical forms are central to identification methods– is not possible for neural nets. Moreover, the proof, based on certain computations with vector fields, is constructive, so it suggests a technique in principle for obtaining parameters from i/o data, totally different from numerical methods based on mismatch minimization; more on that subject later.

A recent variation using exponentials as activation functions was pursued as part of Renee Koplon's just-completed Ph.D. dissertation, in work supported under this project (cf. [43], [42]); the thesis included a discussion of a simple MATLAB implementation of a realization algorithm.

Other "system theoretic" questions can be studied for recurrent nets as well. For instance, the recent work in [8] looked at questions of *observability*, that is, state distinguishability for a known system, as opposed to determination of the systems parameters with a known initial state. This issue is central to the construction of estimators as well as in further understanding minimality. The main result was that observability can be characterized, if one assumes certain conditions on the nonlinearity and on the system, in a manner very analogous to that of the linear case. Recall that for the latter, observability is equivalent to the requirement that there not be any nontrivial A -invariant subspace included in the kernel of C . Surprisingly, the result generalizes in a natural manner, except that one now needs to restrict attention to certain special "coordinate" spaces. More recent work concerns the simultaneous identification of parameters and initial states. This and other questions, such as the characterization of controllability properties, are the subject of ongoing and planned research by the PIs.

Computational Capabilities

The last part of this report deals with questions of what can and cannot be done in the language of theoretical computer science. The topic is the exploration of the *ultimate capabilities* of recurrent nets viewed as *analog computing devices*. This area is a fascinating one, but very difficult to approach. Part of the problem is that, much interesting work notwithstanding, analog computation is hard to model, as difficult questions about precision of data and readout of results are immediately encountered –see the many references in our papers cited below.

In a recent series of papers as well as ongoing research by one of the PIs and a recently-finished graduate student in the project –various aspects are covered in [5], [9], [10], [28], [30], and [36]– we took the point of view that artificial neural nets provide an opportunity to reexamine some of the foundations of analog computation from the new perspective afforded by an extremely simple yet surprisingly rich model, in a context where techniques from dynamical systems theory interact naturally with more standard notions from theoretical computer science. Starting from such a model, this work derived results on deterministic versus nondeterministic computation, and related the study to standard concepts in complexity theory.

One of the most unexpected conclusions was that, at least within the formalism of analog computation described there, recurrent neural nets are a *universal* model, in much the same manner as Turing machines are for classical digital computing. It was shown that no possible "analog computer" –in a sense precisely defined– could ever have more power –again in a precisely defined sense– up to polynomial time speedups. Particularly satisfying theoretically is the fact that the most natural categories for the values of weights correlate perfectly with different natural subclasses of computing devices, and can be summarized in an extremely elegant fashion (see table below).

Another unanticipated –and intriguing– conclusion was that the class NP of nondeterministic polynomial-time digital computation is *not* included in what can be computed in polynomial time

with analog devices (this is proved under standard assumptions of the “ $P \neq NP$ ” type). Thus the solution of combinatorial problems using analog devices may be subject to the same ultimate computational obstructions, for large problem sizes, as with digital computing. There are also many direct consequences of the results that are immediate but nonetheless interesting on their own right. For example, the problem of determining if a dynamical system $x(t+1) = \sigma_n(Ax(t))$ ever reaches an equilibrium point, from a given initial state, is shown to be effectively undecidable (at least for $\sigma = \pi$). (In Hopfield-type nets, when dealing with content-addressable retrieval, the initial state is taken as the “input pattern” and the final state, if there is convergence in finite time, as a class representative.)

Of course, there had been much work before ours concerning the computational power of “neural networks” of various types, starting at least with the classical McCulloch-Pitts work in the early 1940s, and continuing since. The new aspect of our work, which we believe is more appealing and which allows obtaining precise and strong mathematical results, is that we do not assume a separate potentially infinite “storage” device (such as tapes in Turing machines). This memory is instead encoded in real-valued signals inside the system. Related to ours, and to a great extent our initial motivation, was the work by Jordan Pollack on “neuring machines,” but technically our results are completely different, both in their emphasis on efficient computation and in the general results. Perhaps the work closest in spirit is that on real-number-based computation started by Blum, Shub, and Smale. In contrast to that line of work, however, we do not assume that discontinuous, infinite precision “if-then-else” decisions are possible in our model, nor can discrete results be read out of the system through infinite precision measurements. Thus our model is more restricted than the one used by Blum et al. On the other hand, one might reasonably want to restrict the models even more, for instance to account for noise, and this is a topic for further research.

To be a little more explicit, we studied in our work discrete-time recurrent networks (4) with $\sigma = \pi$. (Though more general nonlinearities as well as continuous-time systems are of interest, it is easy to see that using $\sigma = \text{sign}$, as in the McCulloch-Pitts work, would give an even more computational power than finite automata.) Our main results —after suitable precise definitions— are summarized as follows, stated for simplicity in terms of formal language recognition:

Weights	Capability	Polyltime
integer	regular	regular
rational	recursive	(usual) P
real	arbitrary	analog P

More specifically, (1) with integer matrices A , B , and C , one obtains just the power of finite automata; (2) with rational weights recurrent networks are computationally equivalent, up to a constant time speedup, to Turing machines; and (3) with real parameters, all possible languages, recursive or not, are “computable,” but when imposing polynomial-time constraints, the class that results is *analog-P*, for analog polynomial time.

In [32] it is shown how to characterize computations by networks having weights in certain Kolmogorov-complexity definable classes. It is impossible in this brief overview to explain the details of the definitions and constructions in the above series of papers, but it should be emphasized that a central role is played by a Cantor-like representation of internal values, which by virtue of its self-similarity under scaling allows easy updates, and which because of its gaps allows finite-precision decisions and measurements. Another feature is the equivalence between “analog-P” and the class P/poly studied by Karp and Lipton, of nonuniform polynomial-size circuits or equivalently sparse-oracle Turing Machines.

3 Some More Details

This part of the report provides a few more details on some of the topics discussed earlier. Due to lack of space much must be omitted, so references to the literature are given for most fine technical points.

Feedforward Nets

We first introduce some notation and terminology. We let $\mathcal{F}_{n,\sigma,m}$ be the set of functions $f : \mathbf{R}^m \rightarrow \mathbf{R}$ computable by a 1HL net with n hidden units and activation σ , that is, those that can be expressed as in Equation (1), and write $\mathcal{F}_{\sigma,m} := \bigcup_{n \geq 0} \mathcal{F}_{n,\sigma,m}$. The set $\mathcal{F}_{\sigma,m}^p := (\mathcal{F}_{\sigma,m})^p$ of functions whose coordinates are in $\mathcal{F}_{\sigma,m}$ can be thought of as the set of all $f : \mathbf{R}^m \rightarrow \mathbf{R}^p$ computable by 1HL nets with multiple outputs.

Let \mathcal{U} be a set, to be called the *input set*, and let \mathcal{Y} be another set, the *output set*. In the discussion below, for 1HL nets, $\mathcal{U} = \mathbf{R}^m$. To measure discrepancy in outputs, it may be assumed that \mathcal{Y} is a metric space. For simplicity, assume from now on that $\mathcal{Y} = \mathbf{R}$, or \mathcal{Y} is the subset $\{0, 1\}$, if binary data is of interest. A *labeled sample* is a finite set $S = \{(u_1, y_1), \dots, (u_s, y_s)\}$, where $u_1, \dots, u_s \in \mathcal{U}$ and $y_1, \dots, y_s \in \mathcal{Y}$. (The y_i 's are the “labels;” they are *binary* if $y_i \in \{0, 1\}$.) It is assumed that the sample is consistent, that is, $u_i = u_j$ implies $y_i = y_j$. A *classifier* is a function $F : \mathcal{U} \rightarrow \mathcal{Y}$. The *error* of F on the labeled sample S is defined as

$$E(F, S) := \sum_{i=1}^s (F(u_i) - y_i)^2.$$

A set \mathcal{F} of classifiers will be called an *architecture*. Typically, and below,

$$\tilde{\mathcal{F}} = \{F_{\vec{w}} : \mathcal{U} \rightarrow \mathcal{Y}, \vec{w} \in \mathcal{W}\}$$

is a set of functions parameterized by $\vec{w} \in \mathcal{W}$, where r is an integer. An example of an architecture is that of nets with $m=1$ inputs, $p=1$ outputs, and n hidden units, that is, $\mathcal{F}_{n,\sigma} := \mathcal{F}_{n,\sigma,1}$; here the parameter set has dimension $r=3n+1$. The sample S is *loadable* into \mathcal{F} iff

$$\inf_{F \in \mathcal{F}} E(F, S) = 0.$$

Note that for a binary sample S and a binary classifier F , $E(F, S)$ just counts the number of misclassifications, so in the binary case loadability corresponds to being able to obtain exactly the values y_i by suitable choice of $f \in \mathcal{F}$. For continuous-valued y_i 's, loadability means that values arbitrarily close to the desired y_i 's can be obtained.

One may define the *capacity* $c(\mathcal{F})$ of \mathcal{F} via the requirement that:

$$c(\mathcal{F}) \geq \kappa \text{ iff every } S \text{ of cardinality } \kappa \text{ is loadable.}$$

That is, $c(\mathcal{F}) = \infty$ means that all finite S are loadable, and $c(\mathcal{F}) = \kappa < \infty$ means that each S of cardinality $\leq \kappa$ is loadable but some S of cardinality $\kappa + 1$ is not. (Several other natural definitions of capacity measures are possible, in particular the VC dimension mentioned below as well as one in terms of “generic” loadability; see [3].)

Various relations between capacity and number of neurons are known for nets with one hidden layer and Heaviside or sigmoidal activations. It is an easy exercise to show that the results are independent of the input dimension m , for any fixed activation type σ and fixed number of units n .

In the case $m = p = 1$, parameter counts are interesting, so that case will be considered next. Observe that, for 1HL nets with one input, and n hidden units (and $p=1$), there are $3n + 1$ parameters (appearing nonlinearly), though for \mathcal{H} , effectively only $2n + 1$ matter. (In the case of the standard sigmoid, a Jacobian computation shows that these parameters are independent.) For classification purposes, it is routine to consider just the sign of the output of a neural net, and to classify an input according to this sign. Thus one introduces the class $\mathcal{H}(\mathcal{F}_{n,\sigma})$ consisting of all $\{0, 1\}$ -valued functions of the form $\mathcal{H}(f(u))$ with $f \in \mathcal{F}_{n,\sigma}$. Of interest are scaling properties as $n \rightarrow \infty$. Let

$$\text{CLSF}(\sigma) := \liminf_{n \rightarrow \infty} c(\mathcal{F})/r(\mathcal{F})$$

for $\mathcal{F} = \mathcal{H}(\mathcal{F}_{n,\sigma})$ and

$$\text{INTP}(\sigma) := \liminf_{n \rightarrow \infty} c(\mathcal{F})/r(\mathcal{F})$$

for $\mathcal{F} = \mathcal{F}_{n,\sigma}$. These quantities measure (asymptotically) the ratio between the capacity (in the sense just defined) and the number of parameters. For classification, it is shown in [3] that $\text{CLSF}(\mathcal{H}) = 1/3$ and that $\text{CLSF}(\sigma) \geq 2/3$ for any σ which has a nonzero derivative at some point and is *sigmoidal*, meaning technically now that both $\lim_{u \rightarrow -\infty} \sigma(u)$ and $\lim_{u \rightarrow +\infty} \sigma(u)$ exist and are distinct. (It is also shown there that if one allows “direct i/o connections,” that is to say, a linear term added to (1), then $\text{CLSF}(\mathcal{H})$ doubles to $2/3$, which is somewhat surprising.) The sigmoidal bound is best possible, in the sense that for the piecewise linear π one has $\text{CLSF}(\pi) = 2/3$ while it is the case that $\text{CLSF}(\sigma) = \infty$ for even some real-analytic sigmoidals σ . Regarding continuous-valued interpolation, it is shown in the same paper that $\text{INTP}(\mathcal{H}) = 1/3$, and $\text{INTP}(\sigma) \geq 2/3$ if σ is as above and also continuous. It is also shown that $\text{INTP}(\pi) = 2/3$ and that $2/3 \leq \text{INTP}(\sigma_s) \leq 1$ for the standard sigmoid (the proof of the upper bound in this latter case necessitates some algebraic geometry, since the value may be infinite for more arbitrary, even analytic, sigmoidals). These results show the additional power of sigmoidals over Heaviside activations, but they are still very preliminary, and further work is in progress.

Of course, one could use many different approaches to loadability than the idea of numerically minimizing –as in the “backpropagation” approach– a squared error function (or even a variation such as an entropy measure, as also suggested in the neural nets literature). Blum and Rivest showed that *any possible* neural net learning algorithm based on fixed architectures faces severe computational barriers, *assuming the activation being used is \mathcal{H}* . More precisely, they looked at the loading problem: *do there exist weights that satisfy the desired objective?* as a decision question in computational complexity. This issue appears also in the algorithmic part of PAC learning theory. For fixed input dimension and Heaviside activations, loadability becomes essentially a simple linear programming question, but for such activations the problem is NP-hard when scaling with respect to the number of input or output dimensions, as shown in their work. In view of neural network practice, a more relevant result to understanding the ultimate limitations of backpropagation and related techniques would be to determine the complexity of the loading problem when using sigmoidal activations. From an example in [3], it is known that the problem need not be NP-complete, for certain σ . But for the standard sigmoid σ_s the matter is still open. If it were the case that the problem can be solved in polynomial time, this may indicate that perhaps there is no need for complex searches in feedforward learning problems. It is unlikely that this problem will be efficiently decidable in general, but with bounded input dimension it may very well be the case. As an intermediate step, one of the PIs and coworkers were recently able to extend the Blum-Rivest result to π , showing the NP-completeness of the loading problem for nets with two hidden units and activation π .

We now turn to some more details about the uniqueness question, which was mentioned in the above discussion and also in some detail in the Overview part of the report (the remarks made there will not be repeated).

More on Uniqueness

Consider 1HL-nets, indicated by the data $\Sigma := (B, C, b, c_0, \sigma)$ (omitting σ if obvious from the context) and corresponding 1HL-computable functions $f(u) = \text{beh}_\Sigma = C\tilde{\sigma}_n(Bu + b) + c_0$ as in Equation (1). We assume for simplicity that σ is an odd function. Two networks Σ and $\hat{\Sigma}$ are (input/output) *equivalent*, denoted $\Sigma \sim \hat{\Sigma}$, if $\text{beh}_\Sigma = \text{beh}_{\hat{\Sigma}}$ (equality of functions). The question to be studied, then, is: to what extent does $\Sigma \sim \hat{\Sigma}$ imply $\Sigma = \hat{\Sigma}$? We'll say that the function σ satisfies the *independence property* ("IP" from now on) if, for every positive integer l , nonzero real numbers a_1, \dots, a_l , and real numbers b_1, \dots, b_l for which the pairs (a_i, b_i) , $i = 1, \dots, l$ satisfy $(a_i, b_i) \neq \pm(a_j, b_j) \forall i \neq j$, it must hold that the functions

$$1, \sigma(a_1 u + b_1), \dots, \sigma(a_l u + b_l)$$

are linearly independent. The function σ satisfies the *weak independence property* ("WIP") if the above linear independence property is only required to hold for all pairs with $b_i = 0$, $i = 1, \dots, l$.

Let $\Sigma(B, C, b, c_0, \sigma)$ be given, and denote by B_i the i th row of the matrix B and by c_i and b_i the i th entries of C and b respectively. With these notations, $\text{beh}_\Sigma(u) = c_0 + \sum_{i=1}^n c_i \sigma(B_i u + b_i)$. Call Σ *irreducible* if the following properties hold: (1) $c_i \neq 0$ for each $i = 1, \dots, n$; (2) $B_i \neq 0$ for each $i = 1, \dots, n$; and $(B_i, b_i) \neq \pm(B_j, b_j)$ for all $i \neq j$. Given $\Sigma(B, C, b, c_0, \sigma)$, a *sign-flip* operation consists of simultaneously reversing the signs of c_i , B_i , and b_i , for some i . A *node-permutation* consists of interchanging (c_i, B_i, b_i) with (c_j, B_j, b_j) , for some i, j . Two nets Σ and $\hat{\Sigma}$ are *sign-permutation (sp) equivalent* if $n = \hat{n}$ and (B, C, b, c_0) can be transformed into $(\hat{B}, \hat{C}, \hat{b}, \hat{c}_0)$ by means of a finite number of sign-flips and node-permutations. Of course, sp-equivalent nets have the same behavior (since σ has been assumed to be odd). With this terminology, the following holds: *Let σ be odd and satisfy property IP. Assume that Σ and $\hat{\Sigma}$ are both irreducible, and $\Sigma \sim \hat{\Sigma}$. Then, Σ and $\hat{\Sigma}$ are sp-equivalent.* A net Σ has *no offsets* if $b = c_0 = 0$ (the terminology "biases" or "thresholds" is sometimes used instead of offsets. Note, these terms are used for other very different purposes as well). Then, also: *If σ is odd and satisfies WIP, the same statement is true for nets with no offsets.*

Characterizing WIP is essentially trivial: *If σ is a polynomial, WIP does not hold. Conversely, if σ is odd and infinitely differentiable, and if there are an infinite number of nonzero derivatives $\sigma^{(k)}(0)$, then σ satisfies property WIP.* Nets with no offsets appear naturally in signal processing and control applications, as there it is often the case that one requires that $\text{beh}(0) = 0$, that is, the zero input signal causes no effect, corresponding to equilibrium initial states for a controller or filter. Results in this case are closely related to work in the 1970s by Rugh and coworkers and by Boyd and Chua in the early 1980s in control theory. For analytic σ with all derivatives at zero nonvanishing, functions computable by 1HL nets with no offsets are dense on compacts among continuous functions; if σ is odd and all odd derivatives are zero do not vanish, one can so approximate all odd functions.

The more relevant property IP is far more restrictive. For obvious examples of functions not satisfying IP, take $\sigma(x) = e^x$, any periodic function, or any polynomial. However, the most interesting activation for neural network applications is $\sigma(x) = \tanh(x)$, or equivalently after a linear transformation, the standard sigmoid σ_s . In this case, one of the PIs showed in [4] that the IP property, and hence the desired uniqueness statement, does hold. The proof was based on explicit computations for the particular function $\tanh(x)$. An alternative proof is possible, using analytic continuations, and allows a more general result to be established. The idea is that residues can be used to determine the weights. The following is an easy to check sufficient condition for IP to hold (see [17]): *σ is a real-analytic function, and it extends to an analytic function $\sigma : \mathbb{C} \rightarrow \mathbb{C}$ defined on a subset $D \subseteq \mathbb{C}$ of the form: $D = \{|\text{Im } z| \leq \lambda\} \setminus \{z_0, \bar{z}_0\}$ for some $\lambda > 0$, where $\text{Im } z_0 = \lambda$ and z_0 and \bar{z}_0 are singularities, that is, there is a sequence $z_n \rightarrow z_0$ so that $|\sigma(z_n)| \rightarrow \infty$, and similarly for \bar{z}_0 .* It is easy to see that \tanh , as well as most other functions that have appeared in the neural nets experimental literature such as \arctan ,

satisfies the above sufficient property. (Fefferman has recently extended these ideas to several hidden layers, based on the observation that weights in higher layers can be found as iterated accumulation points of residues; his –highly nontrivial– results apply only to the particular case of tanh, however.)

Two Hidden Layers

As discussed in the Overview part, two-hidden-layer nets are necessary for inverse problems and in particular for many control applications. Next we spell out some more details. For the topics treated here, there is no need to define “2HL nets” themselves, but just their behavior. If $\sigma : \mathbf{R} \rightarrow \mathbf{R}$ is any function, and m, p are positive integers, a function computable by a two-hidden layer (“2HL”) net with m inputs and p outputs and activation function σ is by definition one of the type $f \circ \tilde{\sigma}_n \circ g \circ \tilde{\sigma}_l \circ h$, where f, g , and h are affine maps ($n + l$ is then called the number of “hidden units”). A function computable by a 2HL net with *direct input/output connections* is one of the form $Fu + f(u)$, where F is linear and f is computable by a 2HL net. Before discussing control problems, one can understand the necessity of 2HL nets by means of a more abstract type of question. Consider the following *inversion problem*: *Given a continuous function $f : \mathbf{R}^p \rightarrow \mathbf{R}^m$, a compact subset $C \subseteq \mathbf{R}^m$ included in the image of f , and an $\varepsilon > 0$, find a function $\phi : \mathbf{R}^m \rightarrow \mathbf{R}^p$ so that $\|f(\phi(x)) - x\| < \varepsilon$ for all $x \in C$.* One wants to find a ϕ which is computable by a net, as done in global solutions of inverse kinematics problems—in which case the function f is the direct kinematics. Unless we are restricting to a compact domain and f is one-to-one, or if f is very special (e.g. linear), in general discontinuous functions ϕ are needed, so nets with continuous σ cannot be used. However, and this is the interesting part, [2] establishes that nets with just one hidden layer, even if discontinuous σ is allowed, are *not* enough to guarantee the solution of all such problems. On the other hand, it is shown there that nets with two hidden layers (using \mathcal{H} as the activation type) are sufficient, for every possible f, C , and ε . The basic obstruction is due, in essence, to the impossibility of approximating by single-hidden-layer nets the characteristic function of any bounded polytope, while for some (non one-to-one) f the only possible one-sided inverses ϕ must be close to such a characteristic function.

Consider now state-feedback controllers. The objective, given a system $\dot{x} = f(x, u)$ with $f(0, 0)=0$, is to find a stabilizer $u=k(x)$, $k(0)=0$, making $x=0$ a globally asymptotically stable state of the closed-loop system $\dot{x} = f(x, k(x))$. The first remark is that the existence of a smooth stabilizer k is essentially equivalent to the possibility of stabilizing using 1HL nets (with smooth σ). (Thus the simple classes of systems studied in many neurocontrol papers, which are typically feedback-linearizable and hence continuously stabilizable, can be controlled using such 1HL nets.) More precisely, assume that f is twice continuously differentiable, that k is also in C^2 , that the origin is an exponentially stable point for $\dot{x}=f(x, k(x))$, and that K is a compact subset of the domain of stability. Pick any σ which has the property that twice continuously differentiable functions can be approximated uniformly, together with their derivatives, using 1HL nets (most interesting twice-differentiable scalar nonlinearities will do, as shown in the work of Hornik and others). Then, one can conclude that there is also a different k , this one a 1HL net with activation σ , for which exactly the same stabilization property holds. One only needs to show that if $k_n \rightarrow k$ in $C^2(K)$, with all $k_n(0)=0$ —this last property can always be achieved by simply considering $k_n(x) - k_n(0)$ as an approximating sequence—then $\dot{x}=f(x, k_n(x))$ has the origin as an exponentially stable point and K is in the domain of attraction, for all large n . Finally, one knows that there is a neighborhood V of zero, independent of n , where exponential stability will hold, for all n sufficiently large, because $f(x, k_n(x)) = A_n x + g_n(x)$, with $A_n \rightarrow A$ and with $g_n(x)$ being $o(x)$ uniformly on x (this last part uses the fact that σ approximates in $C^2(K)$). Now continuity of solutions on the right-hand side gives the result globally on K .

In general, smooth (or even continuous) stabilizers fail to exist. Thus 1HL-network feedback laws,

with continuous σ , do not provide a rich enough class of controllers. This motivates the search for discontinuous feedback. It is easy to provide examples where 1HL \mathcal{H} -nets will stabilize but no net with continuous activations (hence implementing a continuous feedback) will. More surprisingly, 1HLN feedback laws, even with \mathcal{H} activations, are not in general enough—intuitively, one is again trying to solve inverse problems—but two hidden layer nets using \mathcal{H} (and having direct i/o connections) are always sufficient. More precisely, [2] shows that the weakest possible type of open-loop asymptotic controllability is sufficient to imply the existence of (sampled) controllers built using such two-hidden layer nets, which stabilize on compact subsets of the state space. On the other hand, an example is given there of a system satisfying the asymptotic controllability condition but for which every possible 1HL stabilizer gives rise to a nontrivial periodic orbit.

Linear Systems with Saturated Actuators

The problem of stabilization of linear systems subject to actuator saturation was introduced in the Overview part. As mentioned there, the paper [11] (summary in [39]) provided a construction of controllers that globally stabilize such systems. It was the first complete result for this important kind of control problems, and the only conditions imposed are the obvious necessary ones, namely that no eigenvalues of the uncontrolled system have positive real part and that the standard stabilizability rank condition holds. Interestingly in the present context, it involves neural nets. As far as we know, this result represents the first time that a neural net architecture has been shown to be perfectly suited theoretically to a control problem.

The precise statements are as follows (a much more general result is given in the above references, but for purposes of this report we merely state a simplified version). We first define \mathcal{S} to be the class of all functions σ from \mathbb{R} to \mathbb{R} such that (1) σ is locally Lipschitz, (2) $s\sigma(s) > 0$ whenever $s \neq 0$, (3) σ is differentiable at 0 and $\sigma'(0) > 0$, and (4), $\lim_{|s| \rightarrow \infty} |\sigma'(s)| < 0$. For any finite sequence $\sigma = (\sigma_1, \dots, \sigma_k)$ of functions in \mathcal{S} , we define a set $\mathcal{G}_n(\sigma)$ of functions f from \mathbb{R}^n to \mathbb{R} to be the class of functions $h : \mathbb{R}^n \rightarrow \mathbb{R}$ given by

$$h(x) = a_1\sigma_1(f_1(x)) + a_2\sigma_2(f_2(x)) + \dots + a_k\sigma_k(f_k(x)),$$

where f_1, \dots, f_k are linear functions and a_1, \dots, a_k are nonnegative constants such that $a_1 + \dots + a_k \leq 1$. (In particular, we could of course pick 1HL nets with such small coefficients; more generally, one can have different activations at each node.) Next, for an m -tuple $\mathbf{l} = (l_1, \dots, l_m)$ of nonnegative integers, define $|\mathbf{l}| = l_1 + \dots + l_m$. For a finite sequence $\sigma = (\sigma_1, \dots, \sigma_{|\mathbf{l}|}) = (\sigma_1^1, \dots, \sigma_{l_1}^1, \dots, \sigma_1^m, \dots, \sigma_{l_m}^m)$ of functions in \mathcal{S} , we let $\mathcal{G}_{n,\mathbf{l}}(\sigma)$ denote the set of all functions $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that are of the form (h_1, \dots, h_m) , where $h_i \in \mathcal{G}_n(\sigma_1^i, \dots, \sigma_{l_i}^i)$ for $i = 1, 2, \dots, m$. (It is clear that $\mathcal{G}_{n,\mathbf{l}}(\sigma) = \mathcal{G}_n(\sigma)$ if $m = 1$.)

Let $\delta > 0$. We say that a function $f : [0, +\infty) \rightarrow \mathbb{R}^n$ is *eventually bounded by δ* (and write $|f| \leq_{ev} \delta$), if there exists $T > 0$ such that $|f(t)| \leq \delta$ for all $t \geq T$. Given an n -dimensional system $\mathcal{E} : \dot{x} = f(x)$, we say that \mathcal{E} is *SISS* (small-input small-state) if for every $\varepsilon > 0$ there is a $\delta > 0$ such that, if $e : [0, +\infty) \rightarrow \mathbb{R}^n$ is bounded, measurable, and eventually bounded by δ , then every solution $t \mapsto x(t)$ of $\dot{x} = f(x) + e(t)$ is eventually bounded by ε . For $\Delta > 0, N > 0$, we say that \mathcal{E} is *SISS_L(Δ, N)* if, whenever $0 < \delta \leq \Delta$, it follows that, if $e : [0, +\infty) \rightarrow \mathbb{R}^n$ is bounded, measurable, and eventually bounded by δ , then every solution of $\dot{x} = f(x) + e(t)$ is eventually bounded by $N\delta$. The system is *SISS_L* if it is *SISS_L(Δ, N)* for some $\Delta > 0, N > 0$. For a system $\dot{x} = f(x, u)$, we say that a feedback $u = k(x)$ is *stabilizing* if 0 is a globally asymptotically stable equilibrium of the closed-loop system $\dot{x} = f(x, k(x))$. If, in addition, this closed-loop system is *SISS_L*, then we will say that k is

SISS_L-stabilizing. The *SISS* property, besides being of interest in itself, is essential in allowing a proof by induction. For a square matrix A , we let $\mu(A)$ denote the number of eigenvalues z of A such that $\operatorname{Re} z = 0$ and $\operatorname{Im} z \geq 0$, counting multiplicities.

Our main result is as follows; it asserts the existence of a stabilizer of the above type, and also provides an estimate on the number of “hidden units” needed: *Let Σ be a linear system $\dot{x} = Ax + Bu$ with state space \mathbf{R}^n and input space \mathbf{R}^m . Assume that Σ is asymptotically null-controllable, i.e. that all the eigenvalues of A have nonpositive real parts and all the eigenvalues of the uncontrollable part of A have strictly negative real parts. Let $\mu = \mu(A)$. Let $\sigma = (\sigma_1, \dots, \sigma_\mu)$ be an arbitrary sequence of bounded functions belonging to S . Then there exists an m -tuple $l = (l_1, \dots, l_m)$ of nonnegative integers such that $|l| = \mu$, for which there are SISS_L-stabilizing feedback $u = -kg(x)$ such that (a) $kg \in \mathcal{G}_{n,l}(\sigma)$ and (b) the linearizations of the closed-loop systems are asymptotically stable.*

The paper [34] developed an explicit design concerning a longitudinal flight model of an F-8 aircraft, with saturations on the elevator rate, and tested the obtained controller on the original nonlinear model. (We picked relative small values of these saturations, to analyze our control design under demanding conditions.) We chose the F-8 example since all parameters and typical trim conditions are publicly available, and the model has been often used as a test case for aircraft control designs. The procedure we followed consisted of first linearizing about an operating point and then constructing a globally stabilizing controller for the resulting linearization, with respect to this given trim condition, following the steps in our papers. Finally we proceed to compare the performance of the controller – applied to the original nonlinear airplane and starting reasonably far from the desired operating point – with the “naive” controller that would result from applying a linear feedback law which would stabilize in the absence of saturations. The objective of the work was to show that the calculations in the abstract proofs can indeed be carried out explicitly (though this an extremely simple case compared to the generality of the results in our papers) and moreover, to study the performance of the resulting controller when used for the original nonlinear model (for which there are no guarantees of stability). We consider the results to be encouraging and indicating the usefulness of further work along this direction. In one of the cases simulated, the “naive” design would stabilize as well, though the difference in performance is quite striking. In another case, the “naive” design results in instability, while the one given with our method stabilizes. The overall performance of the system was not satisfactory for practical applications (and in any case, the saturation levels were far to small to be of interest except perhaps in failure modes), but of course this was just a first, simple, effort; much more work needs to be done.

PAC Learning and VC dimension

We now elaborate some more on the topic, already brought up in the Overview part, of learning-theoretic issues. We first review the technical framework. The definitions are as follows (they are standard, but the terminology is changed a bit). An input space \mathcal{U} as well as a collection \mathcal{F} of maps $\mathcal{U} \rightarrow \{0, 1\}$ are given. The set \mathcal{U} is assumed to be countable, or an Euclidean space, and the maps in \mathcal{F} are assumed to be Borel measurable. In addition, mild regularity assumptions are made which insure that all sets appearing below are measurable (details are omitted). Let W be the set of all sequences $w = (u_1, f(u_1)), \dots, (u_s, f(u_s))$ over all $s \geq 1$, $(u_1, \dots, u_s) \in \mathcal{U}^s$, and $f \in \mathcal{F}$. An *identifier* is a map $\varphi : W \rightarrow \mathcal{F}$. The value of φ on a sequence such as the above is denoted as φ_w . The *error* of φ with respect to a probability measure P on \mathcal{U} , an $f \in \mathcal{F}$, and a sequence $(u_1, \dots, u_s) \in \mathcal{U}^s$, is $\operatorname{Err}(P, f, u_1, \dots, u_s) := \operatorname{Prob}[\varphi_w(u) \neq f(u)]$, where the probability is being understood with respect to P . (This is just L^1 distance between two binary functions.) The class \mathcal{F} is (uniformly) *learnable* if there is some identifier φ with the following property: For each $\varepsilon, \delta > 0$ there is some s so that, for every probability P and every $f \in \mathcal{F}$,

$$\operatorname{Prob}[\operatorname{Err}(P, f, u_1, \dots, u_s) > \varepsilon] < \delta$$

(where the probability is being understood with respect to P^s on \mathcal{U}^s). In the learnable case, the function $s(\varepsilon, \delta)$ which provides, for any given ε and δ , the smallest possible s as above, is called the *sample complexity* of the class \mathcal{F} . If learnability holds then $s(\varepsilon, \delta)$ is automatically bounded by a polynomial in $1/\varepsilon$ and $1/\delta$, in fact by $-(c/\varepsilon) \log(\delta\varepsilon)$, where c is a constant that depends only on the class \mathcal{F} . If there is any identifier at all in the above sense, then one can always use the following naive identification procedure: just pick any element f which is consistent with the observed data. This leads computationally to the loading question discussed earlier. In the statistics literature this “naive technique” is a particular case of what is called *empirical risk minimization*.

Checking learnability is in principle a difficult issue, and the introduction of the following combinatorial concept is extremely useful in that regard. A subset $S = \{u_1, \dots, u_s\}$ of \mathcal{U} is said to be *shattered* by the class of binary functions \mathcal{F} if all possible binary labeled samples $\{(u_1, y_1), \dots, (u_s, y_s)\}$ (all $y_i \in \{0, 1\}$) are loadable into \mathcal{F} . The *Vapnik-Chervonenkis dimension* $\text{VC}(\mathcal{F})$ is the supremum (possibly infinite) of the set of integers κ for which there is *some* set S of cardinality κ that can be shattered by \mathcal{F} . (Thus, $\text{VC}(\mathcal{F})$ is at least as large as the capacity $c(\mathcal{F})$ defined earlier.) The main result, due to Blumer, Ehrenfeucht, Haussler, and Warmuth, but closely related to previous results by Vapnik and others is: *The class \mathcal{F} is learnable if and only if $\text{VC}(\mathcal{F}) < \infty$* . So the VC dimension completely characterizes learnability with respect to unknown distributions; in fact, the constant “ c ” in the sample complexity bound given earlier is a simple function of the number $\text{VC}(\mathcal{F})$, independent of \mathcal{F} itself.

In neural net classification problems, where the sign of the output of a net is used to decide membership in a set, one is interested in classes of the type $\mathcal{F} = \mathcal{H}(\mathcal{F}_{n,\sigma,m})$; these are the classifiers implementable by 1HL nets with n hidden units and m inputs (n and m fixed) and activation σ . The best-known result is a by now classical one due to Baum and Haussler, which asserts that $\text{VC}(\mathcal{F}) < dm(1 + \log(n))$, for the choice $\sigma = \mathcal{H}$, where d is a small constant. Thus 1HL nets with threshold activations are learnable in the above sense. In practice, however, one uses differentiable σ , typically σ_s . It is easy to construct examples of sigmoids, even extremely well-behaved ones, for which the VC dimension of the class $\mathcal{H}(\mathcal{F}_{n,\sigma,m})$ is infinite; see [3]. For some time, the question of finiteness of $\text{VC}(\mathcal{F})$ for $\mathcal{F} = \mathcal{H}(\mathcal{F}_{n,\sigma,m})$ and $\sigma = \sigma_s$, and hence (information-theoretic) learnability when using the standard architectures, was open. For bounded weights there were results by White, Haussler and others, but VC bounds depend on the assumed bounds on weights. The question was recently settled theoretically in [33], which showed that indeed $\text{VC}(\mathcal{F}) < \infty$, for activation σ_s . Thus sigmoidal nets appear to have some special properties, vis a vis other possible more general parametric classes of functions, at least from a learnability viewpoint. The paper [33] provided several other results as well, including finiteness of the Pollard-Haussler pseudodimension, which gives also learnability of real-valued as opposed to merely binary functions, and results on “teaching dimension” related to a different paradigm of learning introduced by Goldman and Kearns. The proof VC dimension result in [33] hinged on recent techniques from model theory, combining recent results of Wilkie on order-minimality with work by Laskowski.

Recurrent Nets: Identifiability, Observability

We now turn to recurrent neural networks evolving either in discrete or in continuous time. The dynamics and observation map of such systems can be described by a discrete or continuous time system in the usual sense of control theory, with the special form of the difference or differential equations as in Equation (4), that is, $x^+ (\text{ or } \dot{x}) = \bar{\sigma}(Ax + Bu)$, $y = Cx$. As before, we use the superscripts “ $+$ ” and “ $.$ ” to denote time-shift and time-derivative respectively, and we omit the time arguments t . For the continuous-time case, we assume that the function σ is globally Lipschitz, so that the existence and uniqueness of solutions for the differential equation is guaranteed for all times.

A system of the type in Equation (4) is specified by the data $\Sigma := (A, B, C, \sigma)$; as in the feedforward case, we again omit σ if obvious from the context. (One could add constant terms and consider more general systems x^+ (or \dot{x}) = $\tilde{\sigma}(Ax + Bu + b)$, $y = Cx + c_0$. Such more general systems can be reduced to the above case, by adding a state which has a constant value. Note however that this transformation means that one cannot take the initial state as zero while keeping the above form, and this in fact gives rise to highly nontrivial complications, discussed later.)

As mentioned earlier, the PIs and graduate students have done substantial work on questions of *parameter identifiability* (the possibility of recovering the entries of the matrices A , B , and C from the input/output map $u(\cdot) \mapsto y(\cdot)$ of the system) as well as other related questions such as state-observability. The issue is intimately related with the material discussed earlier on weight determination for feedforward nets, but mathematically the topics are quite different. In particular, some of the technical restrictions regarding σ can be relaxed, because in this context it is often the case that the initial state is an equilibrium state, and this means that property WIP is closer to the problem than property IP. On the other hand, the problem becomes harder because of the dynamics, especially in continuous-time, and tools from nonlinear systems must be employed.

We next provide some more details. Depending on the interpretation (discrete or continuous time), one defines an appropriate behavior beh_Σ , mapping suitable spaces of input functions into spaces of output functions, again in the standard sense of control theory, for any fixed initial state. For instance, in continuous time, one proceeds as follows: For any measurable essentially bounded $u(\cdot) : [0, T] \rightarrow \mathbf{R}^m$, denote by $\phi(t, \xi, u)$ the solution at time t with initial state $x(0) = \xi$; this is defined on $[0, T]$ because of the global Lipschitz assumption. For each input, let $\text{beh}_\Sigma(u)$ be the output function corresponding to the initial state $x(0) = 0$, that is, $\text{beh}_\Sigma(u)(t) := C(\phi(t, 0, u))$. Two recurrent nets Σ and $\hat{\Sigma}$ (necessarily with the same numbers of input and output channels, i.e. with $p = \hat{p}$ and $m = \hat{m}$) are *equivalent*, denoted $\Sigma \sim \hat{\Sigma}$, (in discrete or continuous time, depending on the context) if it holds that $\text{beh}_\Sigma = \text{beh}_{\hat{\Sigma}}$.

Assume from now on that σ is infinitely differentiable around zero, and that it satisfies the following extremely mild nonlinearity assumption:

$$\sigma(0) = 0, \quad \sigma'(0) \neq 0, \quad \sigma''(0) = 0, \quad \sigma^{(q)}(0) \neq 0 \text{ for some } q > 2. \quad (*)$$

(Far less is needed for the results to be quoted, but this is certainly sufficient. For odd analytic functions, we are just asking that σ be nonlinear and nonsingular at zero.) Let $\mathcal{S}(n, m, p)$ denote the set of all recurrent nets $\Sigma(A, B, C, \sigma)$ with fixed n, m, p and any fixed σ as above. Two nets Σ and $\hat{\Sigma}$ in $\mathcal{S}(n, m, p)$ are *sign-permutation equivalent* if there exists a nonsingular matrix T such that

$$T^{-1}AT = \hat{A}, \quad T^{-1}B = \hat{B}, \quad CT = \hat{C},$$

and T has the special form: $T = PD$, where P is a permutation matrix and $D = \text{diag}(\lambda_1, \dots, \lambda_n)$, with each $\lambda_i = \pm 1$. The nets Σ and $\hat{\Sigma}$ are just *permutation equivalent* if the above holds with $D = I$, that is, T is a permutation matrix.

Let $\mathbf{B}^{n,m}$ be the class of $n \times m$ real matrices B for which: $b_{i,j} \neq 0$ for all i, j , and for each $i \neq j$, there exists some k such that $|b_{i,k}| \neq |b_{j,k}|$. For any choice of positive integers n, m, p , denote by $S_{n,m,p}^c$ the set of all triples of matrices (A, B, C) , $A \in R^{n \times n}$, $B \in R^{n \times m}$, $C \in R^{p \times n}$ which are “canonical” (observable and controllable). This is a generic set of triples, in the sense that the entries of those which do not satisfy the property form a proper Zariski-closed subset of $\mathbf{R}^{n^2+nm+np}$. Finally, let:

$$\tilde{\mathcal{S}}(n, m, p) = \left\{ \Sigma(A, B, C, \sigma) \mid B \in \mathbf{B}^{n,m} \text{ and } (A, B, C) \in S_{n,m,p}^c \right\}.$$

Then, in [7], a general result was proved, for a somewhat larger class of systems, which in particular implies:

Assume that σ is odd and satisfies property (*). Then $\Sigma \sim \hat{\Sigma}$ if and only if Σ and $\hat{\Sigma}$ are sign-permutation equivalent.

(An analogous result can be proved when σ is not odd, resulting in just permutation equivalence.) The proof starts with the obvious idea of considering the parameters as extra constant states and computing the observables for the extended systems; this involves considering iterated Lie derivatives of observations under the vector fields defining the system and is a routine approach in nonlinear control. However, and this is the nontrivial part, extracting enough information from these Lie derivatives is not entirely easy, and the proof centers on that issue. The reference cited also included results for the more general class of systems of the Hopfield-type $\dot{x} = -Dx + \tilde{\sigma}(Ax + Bu)$, as well as results showing that $\tilde{\sigma}$ itself can be identified from the black-box data; there is no room here to explain all these results. Also, discrete-time results are now available; see [35]. Moreover, if the activation is analytic, the response to a *single* long-enough input function is theoretically enough for identification of all parameters, just as impulse-responses are used in linear systems theory; this is also explained in [7].

A result on observability was also recently given, in [8]. We start by slightly enlarging the class $B^{n,m}$, to $\mathcal{B}_{n,m}$, the class of those $B \in \mathbf{R}^{n \times m}$ for which no row vanishes and for which the second condition in the definition of $B^{n,m}$ ($\forall i \neq j, \exists k$ s.t. $|b_{i,k}| \neq |b_{j,k}|$) holds. Fix n, m . We denote by \mathcal{S} the set of all recurrent nets for which $B \in \mathcal{B}_{n,m}$ and σ satisfies the property IP. Let $e_i, i = 1, \dots, n$ denote the canonical basis elements in \mathbf{R}^n . A subspace V of the form $V = \text{span} \{e_{i_1}, \dots, e_{i_l}\}$, for some $l > 0$ and some subset of the e_i 's, will be called a *coordinate subspace*; that is, coordinate subspaces are those invariant under all the projections $\pi_i : \mathbf{R}^n \rightarrow \mathbf{R}^n$, $\pi_i e_j = \delta_{ij} e_i$. Sums of coordinate subspaces are again of that form. Thus, for each pair of matrices (A, C) with $A \in \mathbf{R}^{n \times n}$ and $C \in \mathbf{R}^{p \times n}$, there is a unique largest A -invariant coordinate subspace included in $\ker C$; we denote this by $\mathcal{O}_c(A, C)$. One way to compute $\mathcal{O}_c = \mathcal{O}_c(A, C)$ is by the following recursive procedure: $\mathcal{O}_c^0 := \ker C$, $\mathcal{O}_c^{k+1} := \mathcal{O}_c^k \cap A^{-1} \mathcal{O}_c^k \cap \pi_1^{-1} \mathcal{O}_c^k \cap \dots \cap \pi_n^{-1} \mathcal{O}_c^k$, $k = 0, \dots, n-1$, $\mathcal{O}_c := \mathcal{O}_c^n$, which can be implemented by an algorithm which employs a number of elementary algebraic operations which is polynomial on the size of n and m . (Also one could use graph-theoretic computations instead.) The two main results in the above reference (one for discrete and another one for continuous time) are summarized as:

$$\Sigma \in \mathcal{S} \text{ is observable if and only if } \ker A \cap \ker C = \mathcal{O}_c(A, C) = 0.$$

Surprisingly, this is a very simple characterization, and easy to check. Since the corresponding linear system (if σ would have been the identity) is observable if and only if $\mathcal{O}(A, C)$, the largest A -invariant subspace included in $\ker C$, is zero, and since both $\mathcal{O}_c(A, C)$ and $\ker A \cap \ker C$ are subspaces of $\mathcal{O}(A, C)$, one gets in particular that if $\Sigma \in \mathcal{S}$ and the pair of matrices (A, C) is observable then Σ is observable, but the converse does not hold.

Recurrent Nets as Universal Systems Models, and an Algorithmic Approach

Recurrent nets provide universal identification models, in a suitable sense. To be more precise, we may consider continuous- or discrete-time, time-invariant systems Σ : \dot{x} [or x^+] $= f(x, u)$, $y = h(x)$ under standard smoothness assumptions (for instance, $x(t) \in \mathbf{R}^n$, $u(t) \in \mathbf{R}^m$, and $y(t) \in \mathbf{R}^p$ for all t , and f and h are continuously differentiable). It is not hard to prove that, on compacts and for finite time intervals, the behavior of Σ can be approximately simulated by the behavior of a recurrent network, assuming that one uses any activation σ which is universal, in the sense that dilates and translates of σ are dense on continuous functions with the compact-open topology. (If σ is locally Riemann integrable, it is a universal activation if and only if it is not a polynomial, as known from recent results by Leshno and others.)

By approximate simulation we mean as follows. In general, assume given two systems Σ and $\tilde{\Sigma}$ as above, where tildes denote data associated to the second system, and with same number of inputs and outputs (but possibly $\tilde{n} \neq n$). Suppose also given compact subsets $K_1 \subseteq \mathbf{R}^n$ and $K_2 \subseteq \mathbf{R}^m$, as well as an $\varepsilon > 0$ and a $T > 0$. Suppose further (this simplifies definitions, but can be relaxed) that for each initial state $x_0 \in K_1$ and each measurable control $u(\cdot) : [0, T] \rightarrow K_2$ the solution $\phi(t, x_0, u)$ is defined for all $t \in [0, T]$. The system $\tilde{\Sigma}$ simulates Σ on the sets K_1, K_2 in time T and up to accuracy ε if there exist two continuous mappings $\alpha : \mathbf{R}^{\tilde{n}} \rightarrow \mathbf{R}^n$ and $\beta : \mathbf{R}^n \rightarrow \mathbf{R}^{\tilde{m}}$ so that the following property holds: For each $x_0 \in K_1$ and each $u(\cdot) : [0, T] \rightarrow K_2$, denote $x(t) := \phi(t, x_0, u)$ and $\tilde{x}(t) := \tilde{\phi}(t, \beta(x_0), u)$; then this second function is defined for all $t \in [0, T]$, and

$$\|x(t) - \alpha(\tilde{x}(t))\| < \varepsilon, \quad \|h(x(t)) - \tilde{h}(\tilde{x}(t))\| < \varepsilon$$

for all such t . One may ask for more regularity properties of the maps α and β as part of the definition; in any case the maps constructed in the reference given below are at least differentiable. Assume that σ is a universal activation, in the sense defined earlier. Then, for each system Σ and for each K_1, K_2, ε, T as above, there is a σ -system $\tilde{\Sigma}$ that simulates Σ on the sets K_1, K_2 in time T and up to accuracy ε . The proof is not hard, and it involves first simply using universality in order to approximate the right-hand side of the original equation, and then introducing dynamics for the “hidden units” consistently with the equations. This second part requires a little care; for details, see for instance [31]. (Some variations of this result were given earlier and independently in work by Polycarpou and Ioannou, and by Matthews, under more restrictive assumptions and with somewhat different definitions and conclusions.)

Thus, recurrent nets approximate a wide class of nonlinear plants. Note, however, that approximations are only valid on compact subsets of the state space and for finite time, so that many interesting dynamical characteristics are not reflected. Since recurrent nets are being used in many application areas (recall the earlier discussion), this universality property is of some interest. This is analogous to the role of bilinear systems, which had been proposed previously (work by Fliess and by the second PI in the mid-1970s) as universal models. As with bilinear systems, it is obvious that if one imposes extra stability assumptions (“fading memory” type) it will be possible to obtain global approximations, but this is probably not very useful, as stability is often a goal of control rather than an assumption. In any case, the results justify the study of recurrent nets at least as much as they justify the investment in the study of bilinear systems. And the parameter identification result mentioned earlier seems to say that they are from an estimation viewpoint more appealing. (Bilinear realization provides uniqueness of internal parameters only up to $GL(n)$.)

The philosophy of systems modeling using this approach would be to postulate a model of this form (using the fact that the true plant can be so approximated) and then to identify parameters under this assumption. This is fairly much what is done in practice with linear systems; true systems are nonlinear, but linear models often approximate well-enough that the approach makes sense. (Of course, in practice one needs to account for noise, and robust identification techniques are being developed, in the linear case, to deal with the fact that the true plant is not in the class.) In any case, next we suggest a possible approach to numerical identification of a recurrent model for a very special (but still universal) activation. The idea is to use a Fourier-type expansion in the right-hand side of equations. We have recently started looking at the use of $\sigma(x) = \exp(ix)$ as the activation function; let's for now call a recurrent net with this activation function an *exp-system*. This is a natural function to use as we can see from its utilization in Fourier analysis. With the introduction of complex numbers and the use of the exponential, we are able to reduce a complicated nonlinearity in the output to a product. The benefit is that we may use certain input/output pairs to exactly identify the parameters A, B, C, x_0 for a given exp-system in closed form, with no need to solve any nonlinear programming problems. To give the flavor of the algorithm that is possible –assuming to start with that there is no noise in measurements–

suppose we have a system of known dimension n which is modeled by an exp-system, but with unknown values of A, B, C, x_0 . We take the single-input case for simplicity ($m=1$) and allow for complex entries in A, B, C, x_0 .

Assuming we may reset the system to the (mathematically unknown but presumably physically known) initial state and apply any input to obtain the corresponding output, we can identify those parameters using the following procedure. To compute the values of B , we apply inputs of length one with integer values $0, 1, \dots, 2n - 1$. The resulting $2n$ output values $C\sigma(Ax_0), C\sigma(Ax_0 + B), \dots, C\sigma(Ax_0 + (2n - 1)B)$, can be written as

$$\tilde{C}\underline{1} \quad \tilde{C} \operatorname{diag}(\sigma(b_1), \dots, \sigma(b_n))\underline{1}, \dots, \tilde{C} \operatorname{diag}(\sigma(b_1)^{2n-1}, \dots, \sigma(b_n)^{2n-1})\underline{1}$$

where $\underline{1}$ is the column vector $(1, \dots, 1)'$. Here \tilde{C} is some vector, in general different from C if $x_0 \neq 0$. These elements can be seen as the first $2n$ Markov parameters (or impulse response parameters) for an n -dimensional *linear* system whose “ A ” matrix has eigenvalues exactly equal to $\sigma(b_1), \dots, \sigma(b_n)$. These eigenvalues can be obtained by applying linear realization techniques to the sequence. This step of the algorithm is based on Hankel-matrix techniques that are classical in the context of linear recurrences and their multivariable extensions developed in control theory. The method appears in many other areas; for instance in coding theory for decoding BCH codes, and in learning theory for sparse polynomial interpolation. In order to actually compute b_1, \dots, b_n , one would need to apply the “inverse” of σ ; this introduces some complication since there are many complex logarithms, but it can be taken care of by applying a different set of multiples of a different input value. The next step in the identification procedure is to determine the vector C . For this, apply inputs of length 2 with integer values ranging from 0 to $n + 1$ for the first input and 0 to $n - 1$ for the second. This provides a Vandermonde system of equations which can then be solved (again there are complications because of complex logs). Finally, A and x_0 can be obtained. We omit more details here, as in any case not all aspects are yet well understood. But in a just-completed Ph.D. dissertation, in work supported under this project, our graduate student Koplon (cf. [43], [42]) has provided a theoretical framework and programmed a quick MATLAB implementation of the above algorithm, and without any noise in measurements it performs correctly, retrieving an unknown system. This is a major research area, of which we have barely scratched the surface. Much is needed, from the use of least-squares techniques and methods based on Hankel-norm approximation to deal with noise (or model mismatch), to the development of efficient algorithms, to the study of algorithms that use random as opposed to chosen data (in learning terms, online as opposed to query-based).

4 Relevant Publications

1. E. Sontag, H.J. Sussmann, "Backpropagation separates where perceptrons do," *Neural Networks* **4**(1991): 243-249.
2. E. Sontag, "Feedback stabilization using two-hidden-layer nets," *IEEE Trans. Neural Networks* **3**(1992): 981-990.
3. E. Sontag, "Feedforward nets for interpolation and classification," *J. Comp. Syst. Sci.* **45**(1992): 20-48.
4. H.J. Sussmann, "Uniqueness of the weights for minimal feedforward nets with a given input-output map," *Neural Networks* **5** (1992), pp. 589-593.
5. H.T. Siegelmann, E. Sontag, "Turing computability with neural nets," *Appl. Math. Lett.* **4**(6)(1991): 77-80.
6. R. Koplon, E. Sontag, "Linear systems with sign-observations," *SIAM J. Control and Optimization* **31**(1993): 1245-1266.
7. F. Albertini, E. Sontag, "For neural networks, function determines form," *Neural Networks* **6**(1993): 975-990.
8. F. Albertini, E. Sontag, "State observability in recurrent neural networks," *Systems & Control Letters* **22**(1994): 235-244.
9. H.T. Siegelmann, E. Sontag, "On the computational power of neural nets," *J. Comp. Syst. Sci.*, to appear.
10. H.T. Siegelmann, E. Sontag, "Analog computation, neural networks, and circuits," *Theor. Comp. Sci.*, to appear.
11. H.J. Sussmann, E. Sontag, Y. Yang, "A general result on the stabilization of linear systems using bounded controls," *IEEE Trans. Autom. Control*, to appear.
12. R. Koplon, E. Sontag, and M. Hautus, "Observability of linear systems with saturated outputs," *Linear Algebra and Applics.* **205-206**(1994): 909-936.
13. B. DasGupta, H.T. Siegelmann, E. Sontag, "On the complexity of training neural networks with continuous activation functions," *IEEE Trans. Neural Networks*, to appear.
14. M. Donahue, L. Gurvits, C. Darken, and E. Sontag, "Rates of convex approximation in non-Hilbert spaces," submitted.
15. H.J. Sussmann, "On the use of neural networks in the analysis of nonlinear systems: realization, approximation, and feedback control," in *Proc. Congrès Satellite du Congrès Européen de Mathématiques on "Aspects Théoriques des Réseaux de Neurones," Paris, 1992.*
16. H.T. Siegelmann, E. Sontag, and L. Giles "The complexity of language recognition by neural networks," in *Algorithms, Software, Architecture* (J. van Leeuwen, ed), North Holland, Amsterdam, 1992, pp. 329-335.

17. F. Albertini, E. Sontag, and V. Maillot, "Uniqueness of weights for neural networks," in *Artificial Neural Networks for Speech and Vision* (R. Mammone, ed.), Chapman and Hall, London, 1993, pp. 115-125.
18. E. Sontag, "Neural networks for control," in *Essays on Control: Perspectives in the Theory and its Applications* (H.L. Trentelman and J.C. Willems, eds.), Birkhauser, Boston, 1993, pp. 339-380.
19. B. DasGupta, H.T. Siegelmann, E. Sontag, "On the Intractability of Loading Neural Networks," in *Theoretical Advances in Neural Computation and Learning* (Roychowdhury, V. P., Siu K. Y., and Orlitsky A., eds.), Kluwer, Boston, 1994.
20. R. Koplon, E. Sontag, "Quantized systems, saturated measurements, and sign-linear systems," in *Proc. Conf. Inform. Sci. and Systems*, John Hopkins University, 1991, pp. 134-139.
21. R. Koplon, E. Sontag, "Algebraic theory of sign-linear systems," in *Proc. Amer. Automatic Control Conference*, Boston, 1991, pp. 799-804.
22. E. Sontag, "Feedback Stabilization Using Two-Hidden-Layer Nets," in *Proc. Amer. Automatic Control Conference*, Boston, 1991, pp. 815-820.
23. H.J. Sussmann, and Y. Yang, "On the stabilizability of multiple integrators by means of bounded feedback controls," *Proc. 30th I.E.E.E. Conf. Decision and Control*, Brighton, UK (1991), pp. 70-72.
24. W. Maass, G. Schnitger, and E. Sontag, "On the computational power of sigmoid versus boolean threshold circuits," *Proc. of the 32nd Annual IEEE Conference on Foundations of Computer Science*, San Juan, Puerto Rico, 1991, pp. 767-776.
25. R. Koplon, E. Sontag, and M. Hautus, "Output-Saturated Systems," *Proc. Amer. Automatic Control Conference*, Chicago, 1992, pp. 2504-2509.
26. Y. Yang, H.J. Sussmann, E. Sontag, "Stabilization of linear systems with bounded controls," in *Proc. Nonlinear Control Systems Design Symp., Bordeaux, 1992* (M. Fliess, Ed.), IFAC Publications, pp. 15-20.
27. E. Sontag, "Systems combining linearity and saturations, and relations to "neural nets," in *Proc. Nonlinear Control Systems Design Symp., Bordeaux, 1992* (M. Fliess, Ed.), IFAC Publications, pp. 242-247.
28. H.T. Siegelmann, E. Sontag, "Some results on computing with "neural nets," *Proc. IEEE Conf. Decision and Control, Tucson, 1992*, IEEE Publications, 1992, pp. 1476-1481.
29. F. Albertini, E. Sontag, "For neural networks, function determines form," *Proc. IEEE Conf. Decision and Control, Tucson, 1992*, IEEE Publications, 1992, pp. 26-31.
30. H.T. Siegelmann, E. Sontag, "On the computational power of neural nets," in *Proc. Fifth ACM Workshop on Computational Learning Theory*, Pittsburgh, 1992, pp. 440-449.
31. E. Sontag, "Neural nets as systems models and controllers," in *Proc. Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 73-79, Yale University, 1992.
32. J.L. Balcázar, R. Gavaldà, H.T. Siegelmann, E. Sontag, "Some structural complexity aspects of neural computation," in *Proc. 8th Annual IEEE Conf. Structure in Complexity Theory*, San Diego, 1993, pp. 253-265.

33. A. Macintyre and E. Sontag, "Finiteness results for sigmoidal 'neural' networks," in *Proc. 25th Annual Symp. Theory Computing*, San Diego, 1993, pp. 325-334.
34. E. Sontag, Y. Yang, "Stabilization with saturated actuators, a worked example: F-8 longitudinal flight control," *Proc. 1993 IEEE Conf. on Aerospace Control Systems*, Thousand Oaks, CA, 1993, pp. 289-293.
35. F. Albertini, E. Sontag, "Identifiability of discrete-time neural networks," *Proc. European Control Conference*, Groningen, 1993, pp. 460-465.
36. H.T. Siegelmann, E. Sontag, "Analog computation via neural networks," in *Proc. 2nd Israel Symposium on Theory of Computing and Systems (ISTCS93)*, IEEE Computer Society Press, 1993.
37. C. Darken, C., M. Donahue, L. Gurvits, and E. Sontag, "Rate of approximation results motivated by robust neural network learning," in *Proc. Sixth ACM Workshop on Computational Learning Theory*, Santa Cruz, 1993.
38. F. Albertini, E. Sontag, "State observability in recurrent neural networks," *Proc. IEEE Conf. Decision and Control, San Antonio*, 1993, IEEE Publications, 1993, pp. 3706-3707.
39. H.J. Sussmann, E. Sontag, Y. Yang, "A general result on the stabilization of linear systems using bounded controls," *Proc. IEEE Conf. Decision and Control, San Antonio*, 1993, IEEE Publications, 1993, pp. 1802-1807.
40. R. Koplon, E. Sontag, "Sign-linear systems as cascades of automata and continuous variable systems," *Proc. IEEE Conf. Decision and Control, San Antonio*, 1993, IEEE Publications, 1993, pp. 2290-2291.
41. B. DasGupta, H.T. Siegelmann, E. Sontag, "On a learnability question associated to neural networks with continuous activations," *Proc. 7th ACM Conference on Learning Theory*, 1994, pp. 47-56.
42. R. Koplon, E. Sontag, "Techniques for parameter reconstruction in Fourier-Neural recurrent networks," in *Proc. IEEE Conf. Decision and Control, Orlando*, 1994, IEEE Publications, 1994, to appear.

Completed Ph.D. Theses Partially Supported by Grant

43. R. Koplon, *Linear Systems with Constrained Outputs and Transitions*, Ph.D. Dissertation, Rutgers University, 1994.
44. H.T. Siegelmann, *Theoretical Foundations of Recurrent Neural Networks*, Ph.D. Dissertation, Rutgers University, 1993.
45. Y. Yang, *Global Stabilization of Linear Systems with Bounded Feedback*, Ph.D. Dissertation, Rutgers University, 1993.